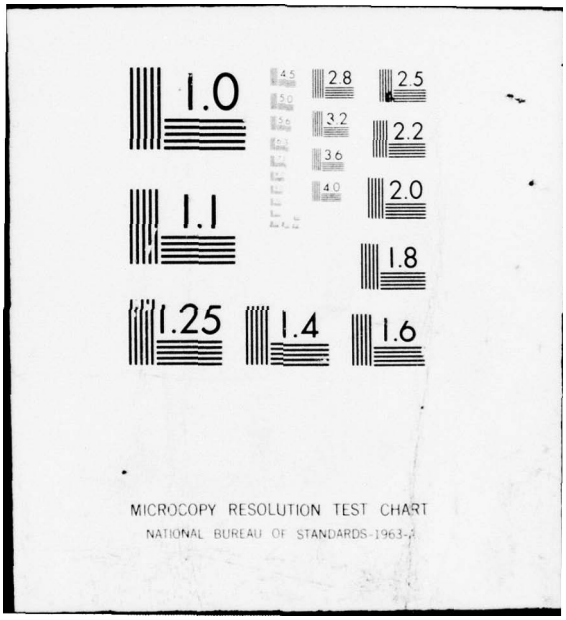
D-A052 040 APPLIED RESEARCH OF CAMBRIDGE LTD (CANADA) F/G 13/13
COMPUTER REPRESENTATION OF THREE-DIMENSIONAL STRUCTURES FOR CAE--ETC(U)
FEB 78 W J MITCHELL, M OLIVERSON DACA88-77-C-0001
NCLASSIFIED CERL-TR-P-86 NL

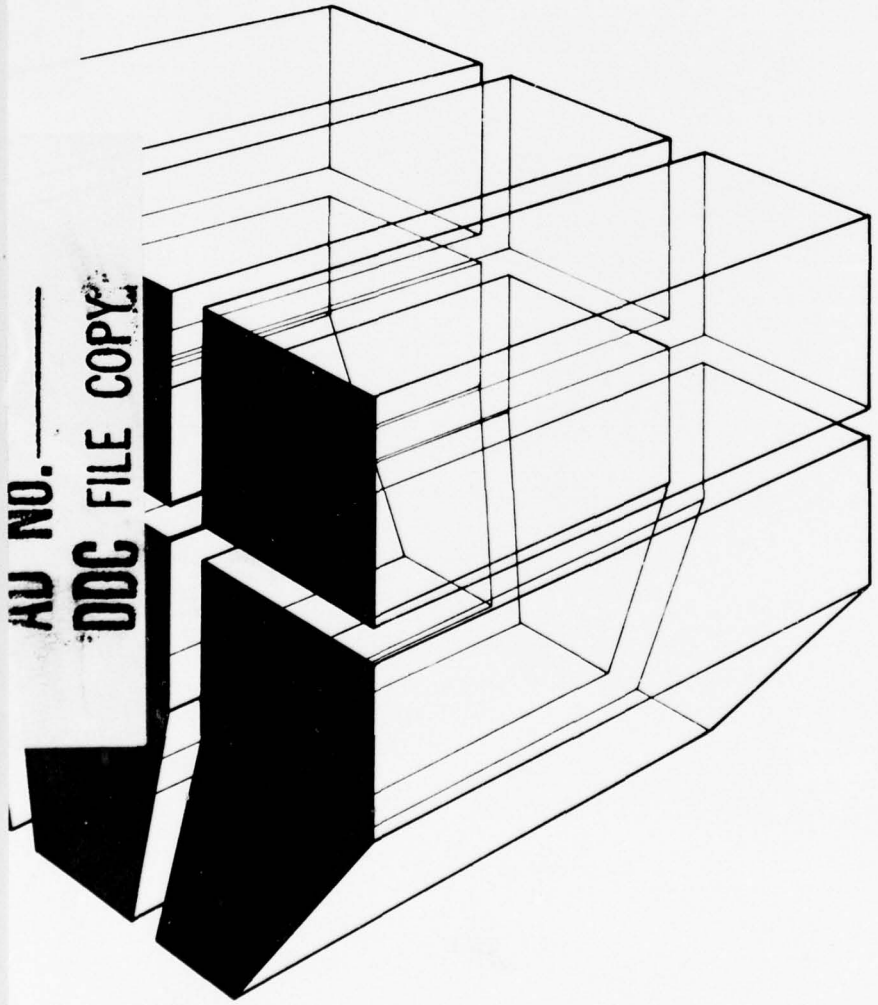
1 OF 3
AD
A052 040


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

construction engineering research laboratory

TECHNICAL REPORT P-86
February 1978
Computer Aided Engineering and Architectural Design System

COMPUTER REPRESENTATION OF THREE-DIMENSIONAL STRUCTURES FOR CAEADS



CERL

The contents of this report are not to be used for advertising, publication, or promotional purposes. Citation of trade names does not constitute an official indorsement or approval of the use of such commercial products. The findings of this report are not to be construed as an official Department of the Army position, unless so designated by other authorized documents.

DESTROY THIS REPORT WHEN IT IS NO LONGER NEEDED
DO NOT RETURN IT TO THE ORIGINATOR

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE

READ INSTRUCTIONS BEFORE COMPLETING FORM

1. REPORT NUMBER: CERL-TR-P-86
2. GOVT ACCESSION NO.
3. RECIPIENT'S CATALOG NUMBER

4. TITLE (and Subtitle): COMPUTER REPRESENTATION OF THREE-DIMENSIONAL STRUCTURES FOR CAEADS.
5. TYPE OF REPORT & PERIOD COVERED: FINAL rept.
6. PERFORMING ORG. REPORT NUMBER

7. AUTHOR(s): William J. Mitchell, Mary Oliverson
8. CONTRACT OR GRANT NUMBER(s): DACA88-77-C-0001 new

9. PERFORMING ORGANIZATION NAME AND ADDRESS: Applied Research of Cambridge Ltd. (Canada), ~~Limited~~ New
410 619
10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS: 4A762731AT41-T1-020

11. CONTROLLING OFFICE NAME AND ADDRESS: CONSTRUCTION ENGINEERING RESEARCH LABORATORY, P.O. Box 4005, Champaign, IL 61820
12. REPORT DATE: Feb~~ruary~~ 1978
13. NUMBER OF PAGES: 195 (12) 195p.

14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)
15. SECURITY CLASS. (of this report): Unclassified
15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release; distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

Copies are obtainable from National Technical Information Service
Alexandria, VA 22151

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

engineering design
architectural design
computer-aided design

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This report presents the findings of a study performed to provide a basis for the selection of a computerized representation of three-dimensional structures for the Computer-Aided Engineering and Architectural Design System (CAEADS).

The project was divided into four interlinked phases:

DD FORM 1 JAN 73 1473 EDITION OF 1 NOV 65 IS OBSOLETE
410 619
JOB
UNCLASSIFIED
SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

Block 20 (continued).

a. Establishment of CAEADS requirements for a building and site description system

b. Identification of criteria against which potential candidate base systems can be evaluated

c. Survey of systems and techniques

d. Evaluation of candidate systems.

It was found that development of a three-dimensional geometric data base system to support CAEADS will be a complex task, but it is feasible. No existing software which exactly matches the requirements could be identified, but a range of feasible implementation strategies which make use of existing available software exists. Strategies based upon the OXSYS system (Applied Research of Cambridge Ltd.), the Evans and Sutherland Design System (Evans and Sutherland Computer Corporation), and GLIDE (Carnegie-Melon University) are most highly recommended.

FOREWORD

This research was conducted for the U.S. Army Construction Engineering Research Laboratory (CERL) under Contract DACA88-77-C-0001 "Services to Provide a Report on Computer Representation of Three-Dimensional Structures" as a part of the Computer-Aided Engineering and Architectural Design System (CAEADS) Project. The CERL Contract Monitor was Dr. William H. Stellhorn.

The research, survey, and report writing were performed by Applied Research of Cambridge (Canada) Limited and its consultants. Personnel directly involved in the study were William J. Mitchell (research) and Mary Oliverson (project manager), with assistance from J. Crispin Gray, Edward M. Hoskins, and Paul N. Richens. Sincere thanks are extended to all who provided information for this study, especially to those who gave up time for personal interviews, and to the CAEADS team at CERL.

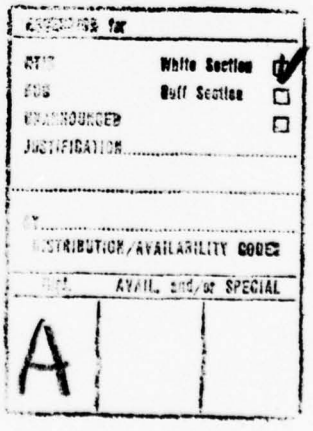
The material in this report is presented only as background information of possible value to Army organizations contemplating similar kinds of software procurement or development. The findings of the report do not indicate any anticipated future action by the Corps of Engineers.

The CAEADS project, of which this study is a part, is under development for the Directorate of Military Construction, Office of the Chief of Engineers (OCE), under Project 4A762731AT41, "Design, Construction and Operations and Maintenance Technology for Military Facilities"; Technical Area T1, "Development of Automated Procedures for Military Construction"; Work Unit 020, "Computer Aided Engineering and Architectural Design System."

The QCDO is 2.10.001, and the OCE Technical Monitor is Mr. Vincent J. Gottschalk.

Col J. E. Hays is Commander and Director of CERL and Dr. L. R. Shaffer is Technical Director. Mr. R. Larson is the Project Manager for the CAEADS Team, and Mr. E. A. Lotz is Assistant Director for Facilities Coordination.

CONTENTS



TABLES

FIGURES

Number	FIGURES (cont'd)	Page
20	Basic Geometric Elements	69
21	Types of Assemblies of Basic Elements	70
22	A Floor Plan and its Dual	71
23	Adjacencies of Faces, Edges, and Vertices	72
24	Point Set Representation of a Shape	74
25	Boolean Description of a Shape	75
26	Boundary Description of a Shape	77
27	Conversions Between Vertices, Faces, and Edges	78
28	Classification of Artifact Geometry	79
29	Approximating the Shapes of Polyhedra	81
30	OXSYS CODEX Indexing	85
31	Indexing of Files	88
32	Conversions of the Ground Model	90
33	Sweep Operation to Create a Cylinder	101
34	Prismatic Object Created by a Project Operation	101
35	Concept of a Parameterized Shape	102
36	Alternative Parameterizations	103
37	Effects of Manipulating an Object Parameterized in Different Ways	104
38	Primitive Parameterized Shapes Provided by BUILD	105
39	Spatial Set Operations	107
40	Use of a Complement Operation to Create a Circulation Space	107
41	Examples of an Automated Component Dimensional Process	109

Number	FIGURES (cont'd)	Page
42	System of Geometric Data Entry Facilities	112
43	Projections of a Washbasin Onto Surfaces of a Surrounding Parallelepiped	116
44	Elevation Produced Using Parallelepiped Method	116
45	Examples of Self-Documentation of E & S Design System Commands	122
46	Typical Property Standard for the Zone Data Property *TD	123
47	Component Data Standard	124
48	CODEX Data for a Component	125
49	Plot of a Catalogued Component	126
50	File Definition and Processing Under CDMS	137
51	COMRADE Command Definition and Evaluation Phases	138
52	Typical STRUDL Input	158

COMPUTER REPRESENTATION
OF THREE-DIMENSIONAL STRUCTURES
FOR CAEADS

1 INTRODUCTION

1.1 BACKGROUND

1.1.1 Overview of CAEADS

The Computer-Aided Engineering and Architectural Design System (CAEADS) project was established to coordinate and advance the introduction and use of computer aids for building design within the Corps of Engineers. The basic stated objective of the system is to improve the efficiency and productivity of the Corps' professional design staff.

The CAEADS project has been divided into two major overlapping phases:

a. Development of stand-alone application programs (e.g., for cost estimating, editing construction specifications, checking spatial requirements, etc. -- see Table 1).

b. Development of an integrated computer-aided building design system capable of embracing all applications within one system (including all possible engineering applications, drafting, and applications developed during the previous phase -- see Table 2).

The pattern which emerges from this approach is one of system growth and evolution; modules are carried from phase to phase until they are replaced by superior new modules. This requires extreme care in the design of the method by which the modules are integrated into the total system.

1.1.2 The User Context

It is anticipated that CAEADS will be employed by all the Corps District Offices and, in many cases, by their consultant architectural and engineering (A/E) firms. Because they have a certain degree of autonomy, the District offices may be expected to exhibit a wide range of variations in the way in which they organize and carry out design, quite independently of regional differences across the United States. Similarly, the A/E firms can be expected to have widely varying procedures, activities, and organizations. Accommodating this diversity

Table 1

Programs Currently Existing or Under Development as a Part of or to Be Incorporated Into CAEADS

TITLE	CAPABILITIES	STATUS
DD Form 1391	1. Justification of requirements 2. Empirical cost estimation 3. Form preparation and distribution 4. Interactive data base searching	Designed, partially implemented. Field testing during FY 78
SEARCH	1. Consistency checking between architectural design criteria and building codes 2. Evaluation of A/E design layouts with regard to design criteria (largely spatial requirements)	Field testing at the Office of the Chief of Engineers (OCE) and two Corps District offices during FY 77
Final Design Cost Estimating System	Preparation of detailed final design (Code C) construction cost estimates	Prototype system developed, operational system designed. Field testing during FY 78
EDITSPEC Phase I Phase II	Based on OCE guide specs: text editing, formatting, and printing automatic editing based on yes/no answers to construction questions	 Operational October 1977 Demonstration operating. Field test in FY 78
BLAST	Estimation of hourly heating and cooling requirements based on building loads and system performance	Operational (not part of CAEADS)
CEUP	Evaluation of utility plans (water, sewage, and electrical) for facility master planning	Various stages of development (not part of CAEADS)
ETIS = Environmental Technical Information Systems	Subsystems: 1. Environ. Impact Computer Systems (EICS) 2. Computer-Aided Environmental Legislative Data System (CELDS) 3. Economic Impact Forecast System (EIFS)	Operational (not part of CAEADS)
LIFE 2	Analysis of pavement design: thickness, earthwork, drainage, frost, cost. Evaluation of maintenance and repair strategies.	Operational (not part of CAEADS)

Table 2

Potential Application Areas for an Integrated CAEADS

	SITE	MASTER PLANNING	PDB AND DD FORM 1391	PROGRAMMING	CONCEPTUAL DESIGN	DETAILED DESIGN	PRODUCTION
CHECKING OF REQUIREMENTS			(DD 1391 processor)	check program against code and guides	check design against functional requirements, code, guides, etc. (SEARCH) →		
DRAFTING	mapping; block perspectives; view analysis; →					working drawings: plans, sections, elevations, details	
REPORTS AND SCHEDULES	various info. to OCE, Div., District, A/E →						schedules: doors windows finishes equipment
COST ESTIMATING L.C.C.	sitework estimates: preliminary → detailed		level A elemental costs →		level B estimated costs based on preliminary bill of materials →		level C: detailed costs (uniformat)
SPECIFICATIONS	site works		program info. →		outline specs. →		constr. specs. (EDITSPEC)
STRUCTURAL			proposal and analysis of alternative systems → prelim. layout and sizing.			automatic sizing & detailing	shop dwgs. schedules etc.
ENERGY		prelim. estimate of requirements sources of supply →			life cycle cost		
MECHANICAL				heat gain/loss by zone, prelim. sizing			
PLUMBING	utilities: alignment sizing (CEUP) detailed design etc.			analysis by zone; prelim. layout and sizing		semi-automatic routing, sizing, selection of network, equipment, etc.	automatic production working drawings; schedules; specs. etc.
ELECTRICAL							
LIGHTING	street lighting; sunlight →						
ACOUSTICS				prelim. analysis →		detailed analysis	
PRODUCT AND MATERIALS SELECTION					generalized type select. →	semi automatic selection →	dwgs specs.
CONSTRUCTION DETAILING	←				semi-automatic detailing →		
MISCELLANEOUS	cut & fill pavement (LIFE 2)			internal traffic analysis →	circulation network analysis		
ENVIRONMENTAL	environmental impact, etc. →						

of users will require that CAEADS be an extremely adaptable and flexible system.

The other general observation that can be made about the user is that he/she may be represented by a typical designer with normal design training in architecture, engineering, estimating, specification writing or an associated profession. He/she cannot be expected to have any knowledge or familiarity with computers -- either their use or their programming. Therefore, CAEADS must be a heavily user-oriented system designed around the skills of a typical engineer.

An average of approximately 80% of Corps design work is carried out by private A/E firms whose designs are checked and approved by the District engineers at various stages. The information flow resulting from this process necessitates that CAEADS be capable of accepting or producing design information at any stage in the design process. If a project were carried out entirely in-house by the District, the whole design procedure could be carried out using the computer (assuming all the necessary components of CAEADS have been implemented). Alternatively, if a project were to be designed by an A/E firm using traditional methods, there would be various exit and reentry points for the computer in active design involvement. CAEADS would still play an important role in multiple checking procedures, but there would be manual design "loops" as shown in Figure 1.

1.1.3 The Construction Context

The buildings designed under the responsibility of the Corps are also widely diversified. The building types emphasized within the Military Construction, Army (MCA) program change from year to year according to troop movements. Averaged over a 10-year period (FY 73 to FY 82), however, the highest dollar volume of planned construction is in such types as barracks and community facilities; hospitals and medical buildings; maintenance, administrative, and training buildings; and waste treatment, utilities, and site works (Table 3).

The size and complexity of the projects also vary widely. At the top of the scale in size are warehouses of nearly 2 million sq ft (185,800 m^2). Health buildings, such as the new, 1,280-bed Walter Reed Hospital,combine size with complexity. Looking to the middle of the scale, an "average" Corps building of medium size and complexity could be represented by an enlisted barracks complex. While these can be very large (over 8,000 personnel), they are

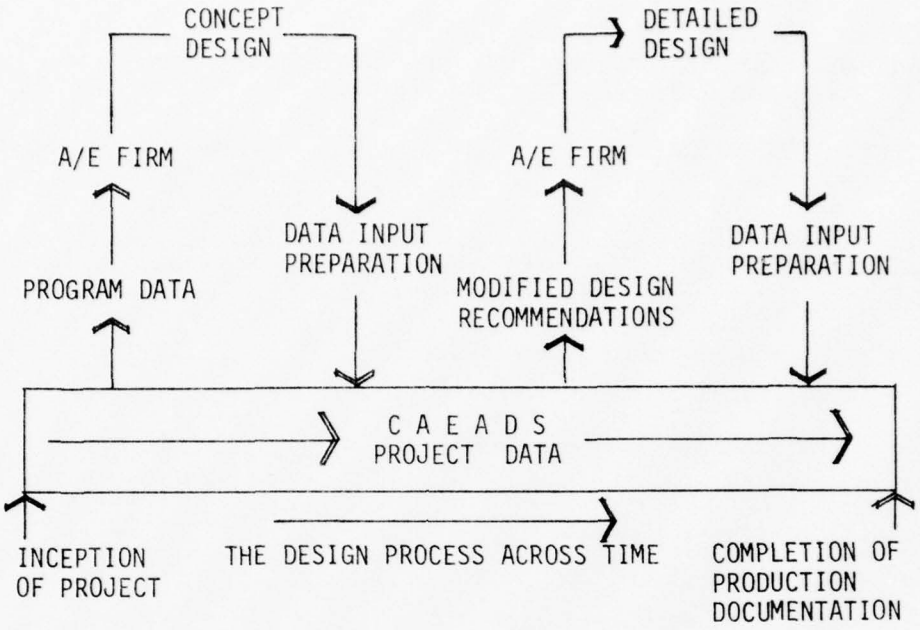


Figure 1 Combining CAEADS with traditional design procedures.

usually broken down into medium-sized blocks.

Even though the Office of the Chief of Engineers (OCE) maintains extensive design guides and standards, the designs and construction methods employed by the Corps demonstrate great diversity; no formalized geometric or construction systems are used. However, it may be assumed that the majority of buildings can be classified into a few categories representing the major traditional methods of construction, with regional variations.

1.2 OBJECTIVES

To integrate a wide range of diverse applications (including undefined and unknown future applications), a computer-aided design system such as CAEADS must be capable of holding a three-dimensional representation of a building and extracting information in any shape or form required by the application programs from that representation.

Table 3

Summary of MCA Data for FY 73 to FY 82
(summarized from OCE computer file "MCA future year system D-File" approximate date February 29, 1976)

CAT.CODE	DESCRIPTION	CURRENT WORKING ESTIMATE (CWE) $ (000)	NO.JOBS
721	Enlisted Barracks	2,221,815	366
740	Community Facilities - Int.	1,230,296	1315
510	Hospitals	699,622	46
214	Maintenance Tank - Automotive	627,720	236
610	Administrative Buildings	587,353	388
171	Training Buildings	530,403	323
831	Sewage & Waste Treatment & Disposal	517,593	210
310	R & D & Test Buildings	378,824	148
730	Community Facilities - Personnel	370,294	217
800	Utilities & Ground Improvements	311,696	210
226	Produc.Fac. - Ammun.Explosiv.Tox.	308,470	31
530	Medical Laboratories	264,164	39
442	Covered Storage	248,158	221
724	Bachelor Officers Quarters	220,490	111
722	Bachelor Housing	215,927	91
421	Ammunition Storage Depot	185,778	35
851	Roads	172,768	167
141	Operational Buildings	142,279	156
211	Aircraft Maintenance Facilities	141,914	62
218	Maintenance Fac's - Misc. Equipment	137,564	63
750	Community Fac's - Exterior	137,012	251
821	Heat & Refrig. Sources	108,557	94

Anything less than a full geometric description automatically precludes the support of certain applications.

In order to proceed with the integration of the existing CAEADS programs, it is necessary to purchase or develop data base software for three-dimensional building description.

The objective of the study documented in this report is to establish the feasibility of, and to provide the basis for the selection and/or development of a computerized representation of three-dimensional structures for use in CAEADS.

1.3 APPROACH

The approach used in this study may be explained by the diagram shown in Figure 2. The work for this project was divided into four heavily interlinked, overlapping phases, starting with the examination of CAEADS requirements for building and site

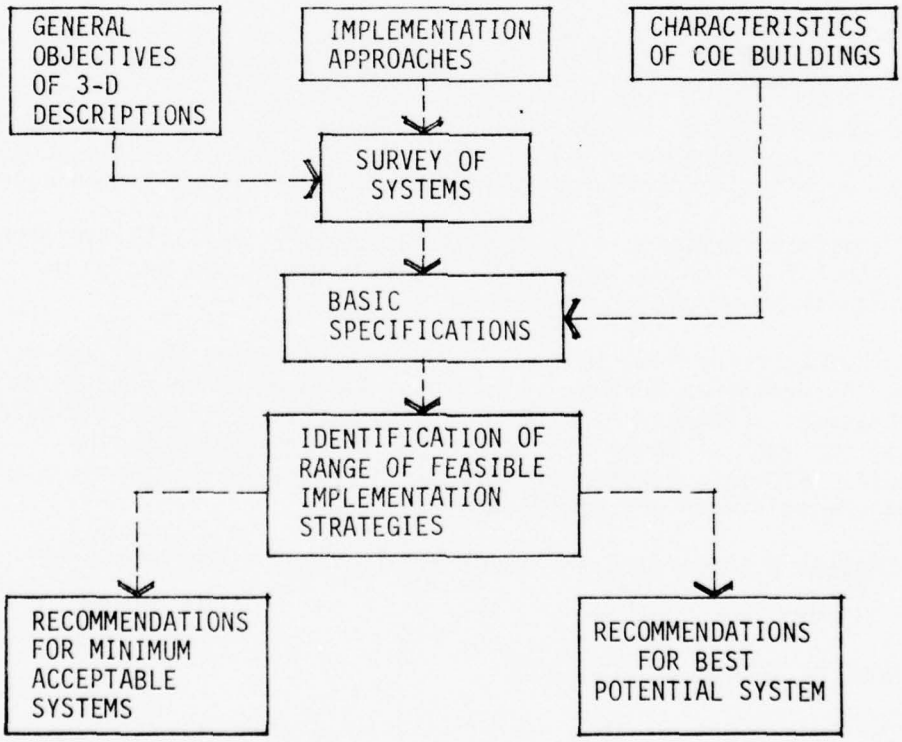


Figure 2. Approach to 3-D description system selection/development.

description and the likely applications to be integrated into the system. The first phase also included research and discussion of various techniques for three-dimensional description.

The second phase of the project was concerned with the identification of minimum- and medium-range requirements, and consequent criteria by which existing candidate description systems could be evaluated.

Existing data base and three-dimensional description systems were then surveyed in detail. This survey entailed 2 weeks of travel throughout the United States and 2 weeks in Great Britain meeting with the originators of candidate systems. For comparison purposes, information was also obtained on drafting and mapping systems, engineering application systems, and many others (see Chapter 4, section 4.1).

The final phase of the project was concerned with evaluation of the systems surveyed.

1.4 REPORT ORGANIZATION

Chapters 2 and 3 contain detailed analyses of the conceptual and technical issues which must be addressed in design and implementation of the CAEADS data base system. In Chapter 2, the general structure of a system suitable for handling a comprehensive three-dimensional building description is analyzed. Chapter 3 discusses the characteristics of the geometric model to be stored in the system.

Chapter 4 surveys a wide range of currently available software that either is potentially useful for implementing the system or might appear to be useful.

Drawing on the information presented in Chapters 2, 3, and 4, Chapter 5 identifies a range of feasible implementation strategies. Criteria for software evaluation are then summarized, and candidate systems for each of these strategies are evaluated against the criteria. Finally, recommendations on implementation strategy and software acquisition are presented.

Chapter 6 summarizes the study findings and recommendations.

1.5 MODE OF TECHNOLOGY TRANSFER

The information in this report will be used in the development of CAEADS and will not impact directly on existing Corps of Engineers or Army documentation. The technology transfer for CAEADS will be accomplished in accordance with techniques for computer-assisted programs as defined in appropriate Army regulations.

2 GENERAL STRUCTURE OF A BUILDING DESCRIPTION DATA BASE SYSTEM

2.1 THE NEED FOR A DATA BASE

2.1.1 Traditional Design Processes

In a traditional building design process (Figure 3), the building description developed as the process unfolds is stored in the form of marks on paper, such as plans, elevations, sections, schedules, bills of materials, etc. This traditional approach has the following disadvantages:

a. Nonintegration. Different documents store different types of information: shapes of objects and their locations in a horizontal plane are recorded in elevation or section, cost data are recorded in schedules, and so on. Thus, it is often necessary to correlate data from several different sources in order to execute a design task.

b. Redundancy. The same information often appears in several different forms in several different places. For example, a room might be drawn in several different plans, produced at different scales for different purposes. This introduces the possibility of inconsistency between different representations and makes alteration a laborious process.

c. Fixed Views. Drawings give a limited set of fixed views of a building. It is more desirable to have facilities which produce sections along any arbitrary plane, perspectives from any arbitrary viewpoint, etc., as required.

d. Coordination Problems. On any reasonably large project, design is carried out by a team rather than an individual. The members of the team usually work on their own copies of the master drawings and are often unaware of the actions of other members, resulting in a lack of coordination and consistency.

e. Obsolete Data. Coordination problems are traditionally resolved by periodically collating data onto a new set of master drawings, which then serve as the basic reference for further work. As this is a slow and expensive process, it is not undertaken very frequently, and the data on the master drawings are therefore often obsolete.

f. Inefficient Data Processing. A large amount of design staff time is spent performing data processing rather than decision-making tasks, e.g., copying, changing scale or format of drawings, taking off quantities, annotating, tabulating, updating, checking for accuracy, etc. Manual performance of these tasks is slow, expensive,

and a major source of errors. Furthermore, it reduces the amount of time available for actual design and evaluation work.

g. Non-Machine-Readability. Data stored in the form of drawings, etc., cannot be operated upon directly by computer (e.g., to perform engineering analyses or to generate new displays or reports). An expensive, time-consuming, and error-prone data preparation and input step is required.

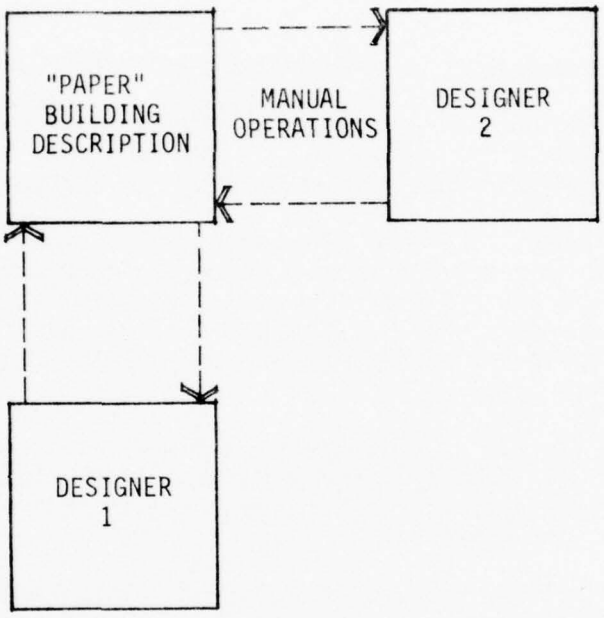


Figure 3. Data flow in a traditional architectural design process.

These disadvantages may not appear particularly significant in the context of a small private architectural practice, where a few professionals may work closely on a relatively small project which is completed in a relatively short time. However, in an organization such as the Corps of Engineers, which deals with a large number of diverse projects -- some of great size and complexity, often requiring coordination of large and perhaps geographically scattered

project teams -- they assume major significance. Any substantial step towards overcoming these disadvantages would have important benefits: decreased design time and cost, more effective project cost and quality control, and simplification and rationalization of project management, review, and evaluation.

The introduction of computer aids has been largely motivated by a desire to gain these benefits, although they have often proved surprisingly elusive in practice, especially in architectural applications. It is useful to examine the history of computer-aided building design to analyze why this has been so.

2.1.2 Introduction of Discrete Application Programs

For most organizations, the first step towards computer-aided design is implementing discrete application programs to perform specific data processing, analysis, or limited design synthesis operations within the overall framework of an essentially manual design process (Figure 4). Implementation of such programs requires relatively low investment with minimum disruption of established work patterns, combined with minimal user training and resistance.

As the employment of application programs in the field increases, rationalizing and standardizing formats, documentation, and style become worthwhile. Careful attention is paid to issues of portability and to the development of programming aids such as special extensions to FORTRAN and high level problem-oriented languages.

Systems such as STRESS, ICES, and the work of APEC[1] are examples of this second stage of automation; they associate discrete application programs by a common input language and offer a degree of portability.

The success of discrete application programs in practice depends upon the ratio between the cost of data preparation, input, and processing, and the benefits achieved by making a run. For certain engineering applications, particularly in the structural field, the cost/benefit ratio has proven sufficiently favorable to justify wide usage. However, this has not been the case for a broad spectrum of architectural applications, particularly those requiring input of a large amount of geometric data (e.g., perspective production). In many cases, data preparation and input costs have been sufficiently high to substantially detract from (or in some cases even to outweigh) the benefits achieved.

---

1 Abstracts of Computer Programs, (Automated Procedures for Engineering Consultants (APEC), 1972)

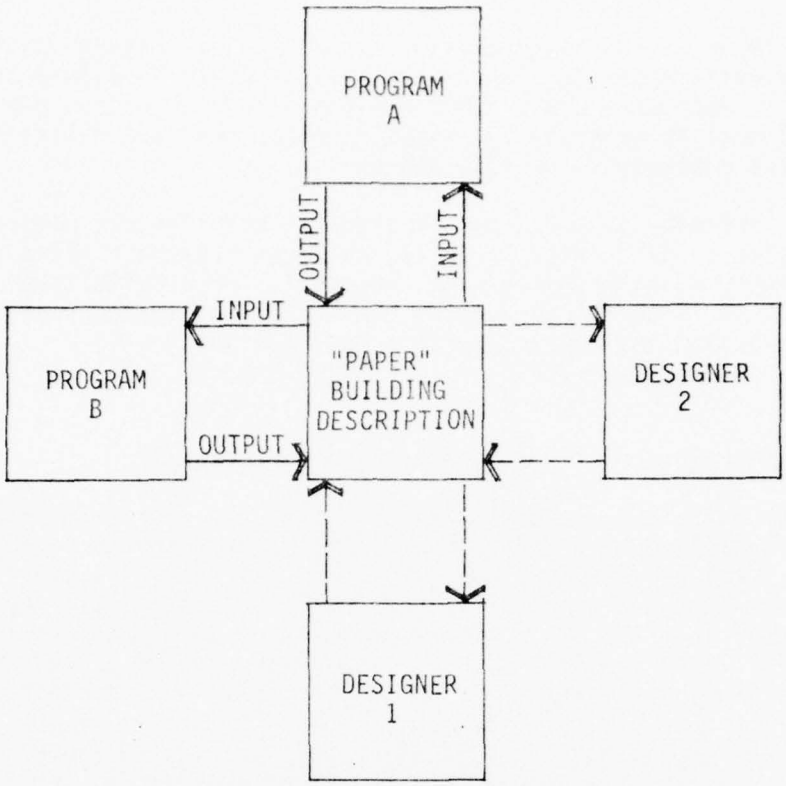


Figure 4. Use of discrete application programs.

A second important limitation is that the use of discrete application programs does not alleviate problems arising from nonintegration, redundancy, and obsolescence of data. Indeed, it may even exacerbate these problems by introducing even more data formats and versions.

2.1.3 Use of a Single Data Set for Several Applications

An obvious way of overcoming some of the disadvantages of discrete application programs is to integrate two or more programs by designing them to operate upon the same input data set (Figure 5). This spreads the cost of input over several applications and reduces the proliferation of different versions of data. By combining several application programs with some kind of monitor and a data editor, a simple interactive integrated system can be developed. The Corps' SEARCH system, which generates a variety of different types of evaluations from a single stored description of building form, is

an example of this approach. Further examples are the Cambridge Environmental Model[2] which performs daylight, artificial lighting, thermal, and acoustical evaluations, and the PACE system[3] developed at the University of Strathclyde in Scotland, which performs schematic environmental, circulation, and cost analyses.

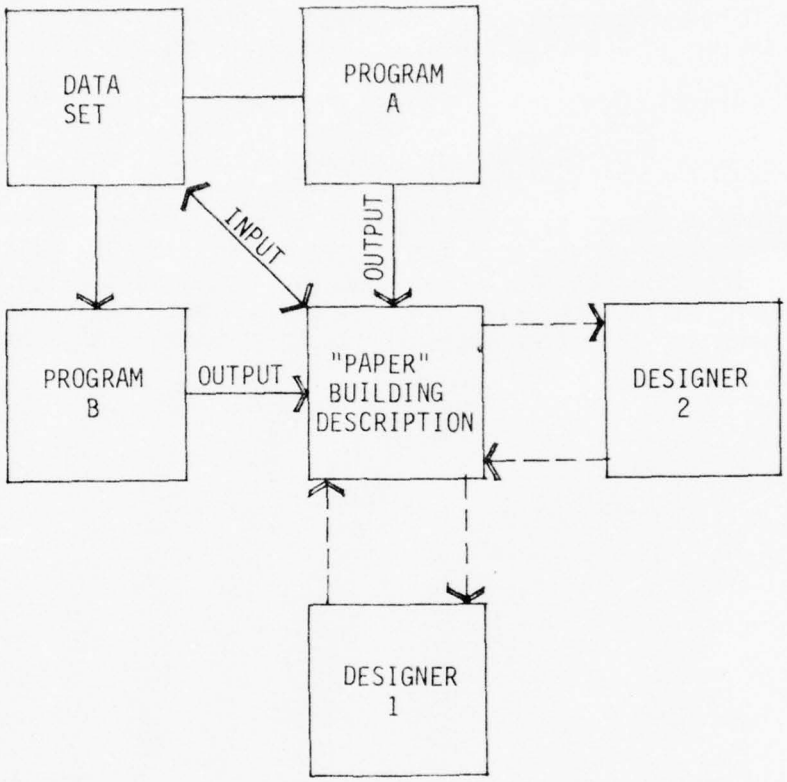


Figure 5. Use of a single data set for several applications.

This approach, although an improvement, also has severe intrinsic limitations. It only works well where the types of analyses are in some way similar, so that the algorithms employed can all make efficient use of the same data structure. This is unlikely to be the case in an arbitrary collection of diverse applications such as beam sizing, pedestrian circulation analysis, perspective production, and detailed thermal analysis.

[2] D. Hawkes and R. Stibbs, The Environmental Evaluation of Buildings Working Papers 15, 27, 28 (University of Cambridge Centre for Land Use and Built Form Studies, 1970)

[3] T. Maver, "PACE 1: Program for Building Appraisal," Architects Journal (The Architectural Press, April 23, 1973)

Future advances in applied mathematics may possibly alleviate this difficulty by demonstrating that apparently unrelated problems are in fact special cases of some generic problem and that common algorithms and data structures may be employed. The success of linear programming methods in operations research and that of finite element methods in structural engineering exemplify this type of development towards generality. However, there is no indication that a theory for dealing with the full range of architectural and engineering design problems is likely to emerge in the near future.

2.1.4 Connection of Programs in Sequence

Another approach to integration is to connect several related programs in sequence, so that a data set output from one is used directly as input to the next, thus reducing input costs (Figure 6). For example, thermal analysis, mechanical equipment sizing, and costing programs are often sequentially integrated in this way.

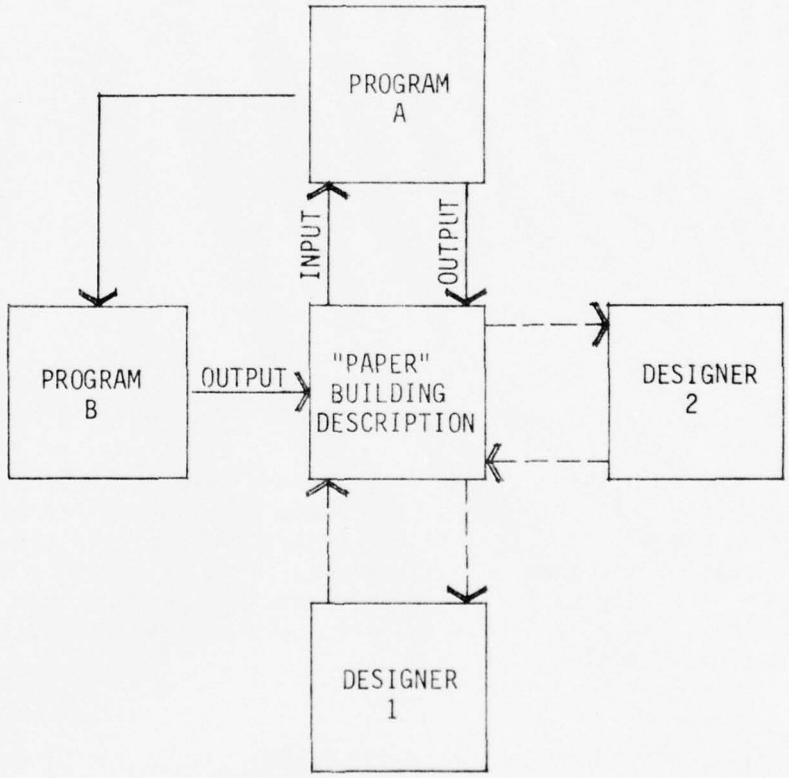


Figure 6. Connection of programs in sequence.

This approach can be effective where there is a very clearly defined linear sequence of steps in a design process. However, because it is generally unsatisfactory to impose a rigid linear sequence over the architectural design process, the potential scope of this method of integration is severely limited.

2.1.5 Use of a Comprehensive Data Base

It can be seen that the scope of the foregoing methods of integrating discrete application programs to reduce input costs and rationalize data flow is limited. The only proven way of achieving total integration is by organizing a computer-aided design system around a comprehensive data base (Figure 7).

Using this approach, a multi-indexed structure of data, semi-permanently stored in computer memory, replaces conventional paper building descriptions as the primary and definitive description of a design. Data need only be entered once into this data base and may subsequently be operated upon by all design application programs. Drawings, printed reports, and machine-readable data sets formatted in specified ways may be generated as required through use of report-generation facilities.

From the designer's point of view, an interactive graphics work station consisting of a graphics display screen, keyboard, and digitizer tablet replaces the drawing board. The necessity for manual data handling is minimized, while the opportunity for application of computer processing is maximized.

The comprehensive data base approach provides an opportunity to effectively overcome the problems of nonintegration, redundancy, fixed views, coordination problems, obsolete data, inefficient data processing, and non-machine-readability of data, as discussed in greater detail in the following sections.

2.2 TECHNIQUES OF DATA BASE IMPLEMENTATION

2.2.1 The Need for Special Data Base Implementation Software

The simplest type of building description data base is a sequential file, stored on disk, which is read into an in-core data structure when required. A slightly more elaborate arrangement is to have several such files and to read them in with different program overlays. A typical example of this approach was the pioneering

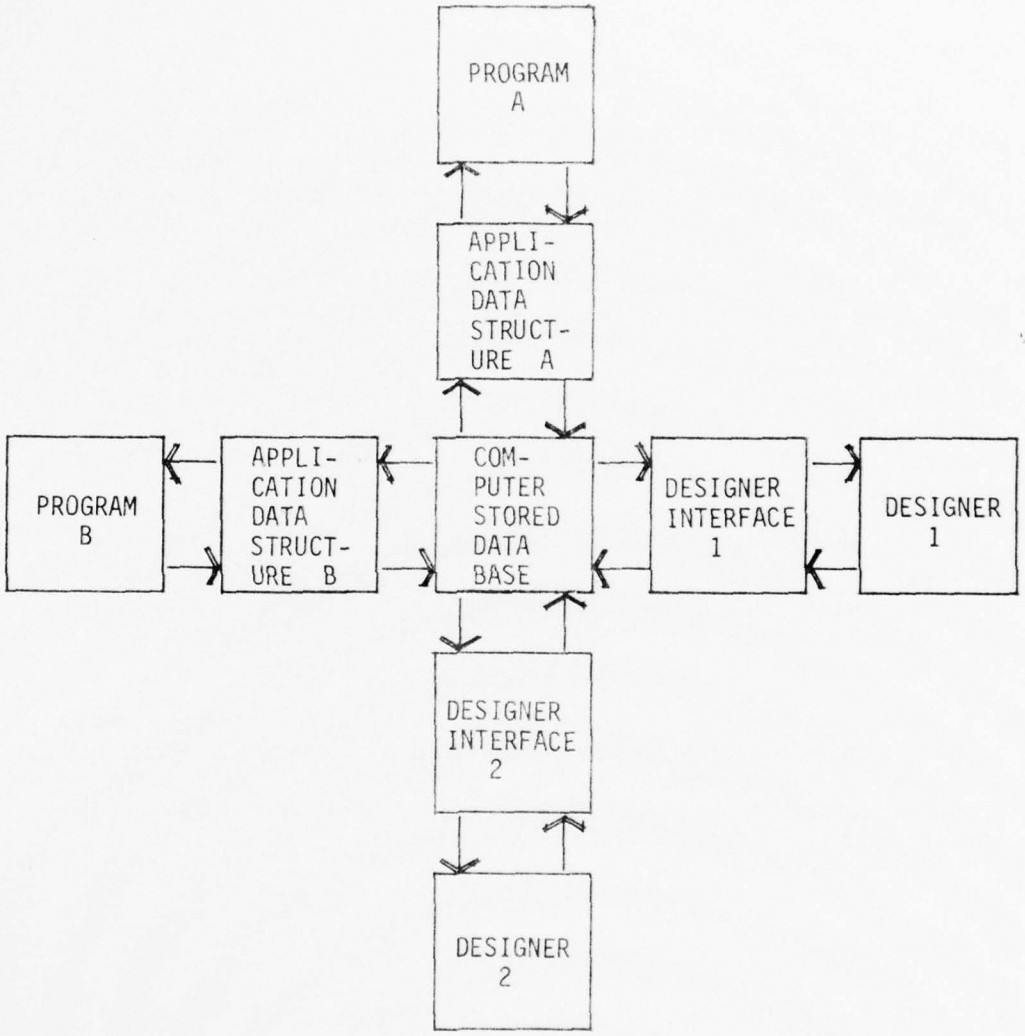


Figure 7. Use of a comprehensive computer-stored data base for building description.

CADS system developed at UCLA.[4] CADS, implemented in the Euler language, stored building descriptions in core using Euler's powerful data structuring facilities. It wrote these data onto disk files at the end of each session, and read the files back at the beginning of the next session. However, this was an academic exercise, and comprehensive building description of a realistic scale and complexity was found to be too large and complicated to be handled in this way.

What is needed for computer-aided design is some kind of structured file which will facilitate organization and accessing of large quantities of data, together with some flexible and efficient technique for organizing the continual flow of data back and forth between core and disk. In response to these types of needs (which are not restricted to computer-aided design) a variety of data base management software concepts and systems have evolved. Figure 8 illustrates the "family tree" of these concepts, which are briefly discussed below. Detailed discussion of particular systems will be found in Chapter 4.

2.2.2 Data Base Management Systems

One line of development began with COBOL and the processing of large sequential files in business applications. As applications became more sophisticated, sequential files proved increasingly inadequate, and systems were developed for handling various types of structured files, particularly hierarchical and ring structures (Figure 9). The development of interactive computer graphics systems led to the study of ring-structured data bases and the development of special software to create and manipulate this type of structure.[5] Around 1966/67, the CODASYL Data Base Task Group (DBTG) began to define concepts, terminology, and standards for generalized data base management systems; a final report was published in 1971.[6] The DBTG proposal sees data bases essentially as networks in which records are vertices and pointers are edges. A data definition language (DDL) is used to define the particular network data structure to be used, and a data manipulation language (DML) is used by the application programmer to access and operate upon records stored in this structure. The separation of DDL and DML enables the physical organization of data in storage to be altered without requiring

4 W.J. Mitchell, "Vitruvius Computatus," in D. Hawkes (ed.) Models and Systems in Architecture and Building, (The Construction Press, 1975).

5 J. A. Hamilton, A Survey of Data Structures for Interactive Graphics, Rand Corporation Memorandum RM-6145-ARPA (April 1970).

6 Data Base Task Group (DBTG) of CODASYL Programming Language Committee Report (April 1971).

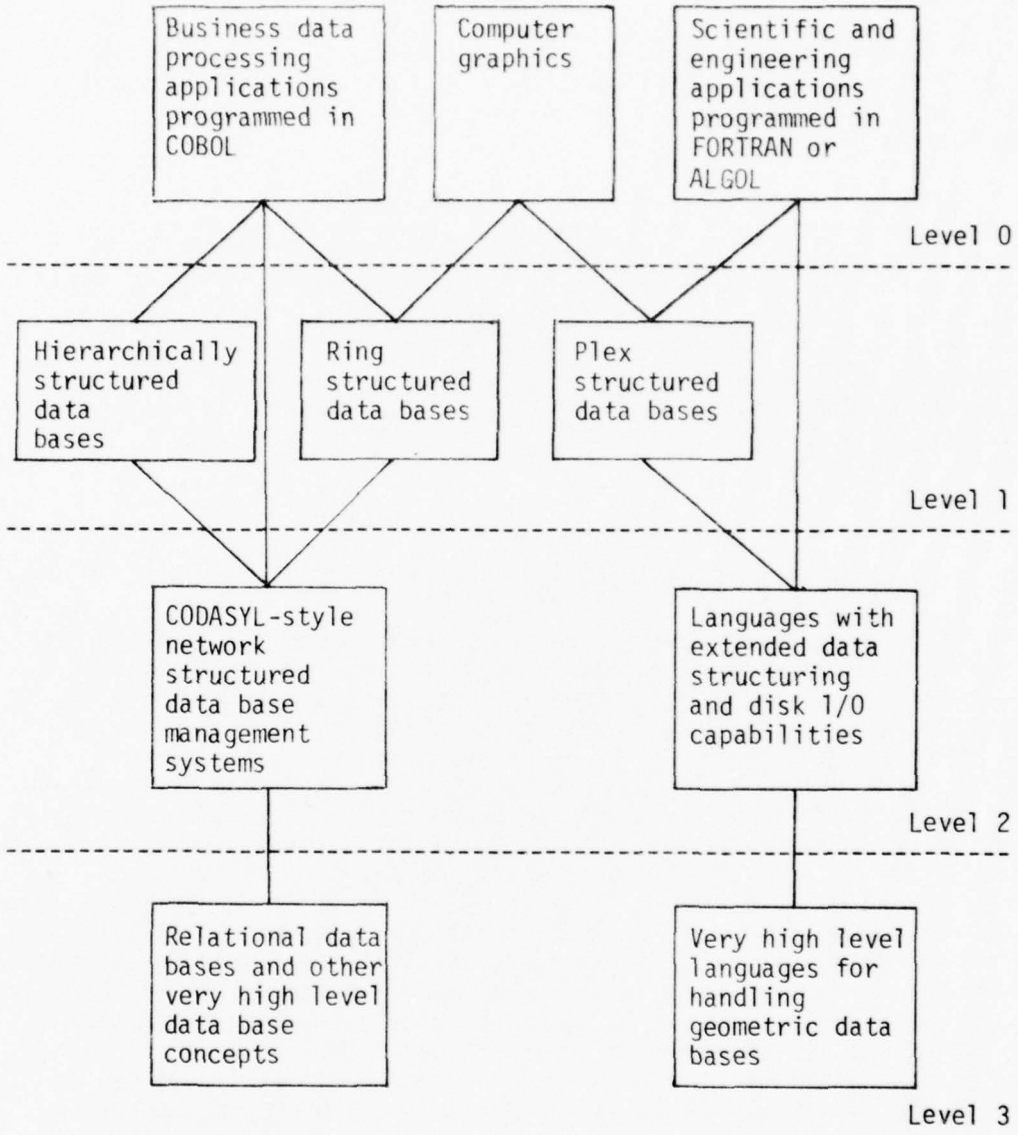
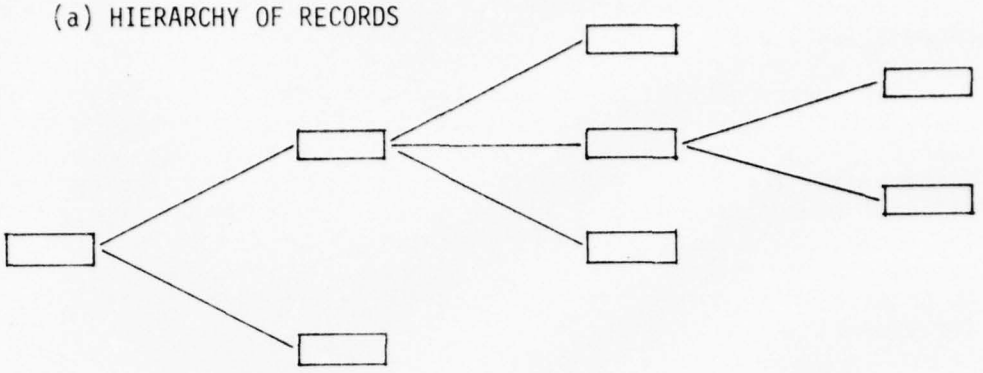
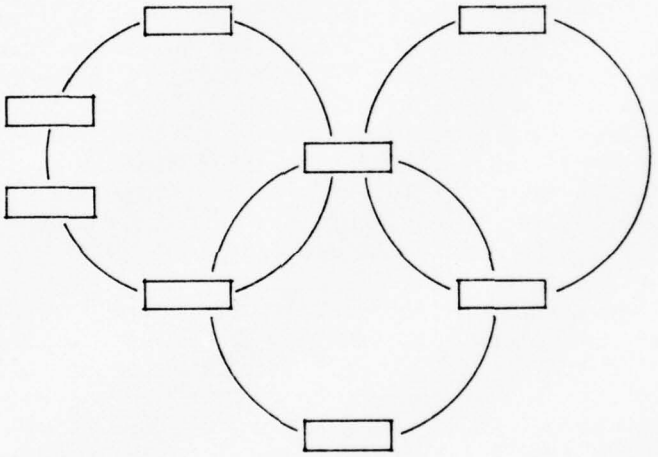


Figure 8. Development of data base management technology.

(a) HIERARCHY OF RECORDS



(b) RINGS OF RECORDS



(c) NETWORK OF RECORDS

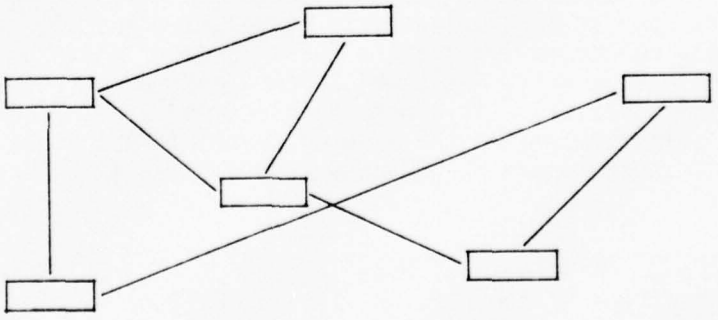


Figure 9. Structures.

rewriting of applications software. However, it tends to introduce computational overhead costs.

The theory of data bases has continued to develop following the early CODASYL work. Perhaps the most significant contribution has been E. F. Codd's[7] concept of a relational data base. Based upon the mathematical theory of relations, this concept represents a very significant conceptual advance, giving numerous important insights into the fundamental nature of data and data structures. In relation to computer-aided design, however, relational data bases must at present be considered a research topic rather than a practical possibility for production implementation.

2.2.3 Programming Languages

Parallel to the CODASYL line of development has been another (not entirely separate) stream beginning with the application of languages such as FORTRAN and ALGOL to the solution of scientific and engineering problems. When the computations undertaken began to grow in scale, the data structuring and disk input/output facilities of these languages were found to be inadequate for engineering problems that involve processing large quantities of highly structured data. Consequently, efforts have been made to enhance their capabilities in these directions, either by extending existing languages, or by developing new languages.

One of the most significant early improvements was the concept of a plex data structure introduced by D. T. Ross in the context of the AED engineering design system.[8] A plex is essentially a self-describing, variable size record, in contrast to the records of fixed size and format which are usually employed in data definition languages. This flexibility has proven to be an important advantage in sophisticated computer-aided design applications.

A large number of programming languages have now been developed which incorporate many of the data structuring and disk input/output facilities needed to implement a building description data base. Examples include PL/1, ALGOL 68, PASCAL, and EULER (see Chapter 4 for further details). At a more specialized level, languages specifically oriented towards the task of geometric description can be developed, such as the GLIDE language developed at Carnegie-Mellon University.

---

[7] E. F. Codd, "A Relational Model of Data for Large Shared Data Banks," Communications of the ACM, Vol 13, No. 6 (June 1970)

[8] D. T. Ross, "The AED Free Storage Package," Communications of the ACM, Vol 10, No. 8 (August 1967)

2.2.4 The Concept of System Levels

Data base implementation software can be classified according to its level. Low level software is used to describe elementary operations or basic units of data (bytes, records, etc.). It is often closely related to specific hardware. High level software allows a programmer or user to express data base operations very concisely, in a convenient and natural language. A single high level command may result in execution of a great many low level operations. High level software is normally closely related to a specific application area.

Ambitious data base systems such as proposed for CAEADS are normally built in a hierarchy of software levels. Each level of software is used as a tool for implementation of the level above. This vastly simplifies the implementation task, and produces a more robust and portable system.

Figure 10 distinguishes six potential levels of software which might be employed in implementing a building description data base system. Examples of well-known existing software, classified according to level, are shown in the left-hand column (descriptions of these examples will be found in Chapter 4).

At the lowest level are general-purpose programming languages such as COBOL, FORTRAN, and ALGOL. It would be exceedingly difficult, if not impossible, to implement the proposed CAEADS data base using only facilities at this level.

At the next level are various extensions of these languages, which provide disk input/output, extended data structuring, and other facilities.

At level 3 are the generalized data base management systems and very high level languages introduced in section 2.2.1. These provide very powerful facilities for rapid system implementation.

Level 4 includes systems specifically oriented towards the description of complex geometric objects. These systems incorporate algorithms for performing tasks of computational geometry. These algorithms are transparent to the user and can be invoked by high level statements. Geometric description systems at this level are general enough to support a wide range of computer-aided design applications (e.g., building, ship, and mechanical part design).

General building description software, at level 5, incorporates algorithms that are specific to building description applications. These systems are more precisely tailored to architectural applications than general geometric modeling systems, but are correspondingly less broad in their range of application. The highest level systems

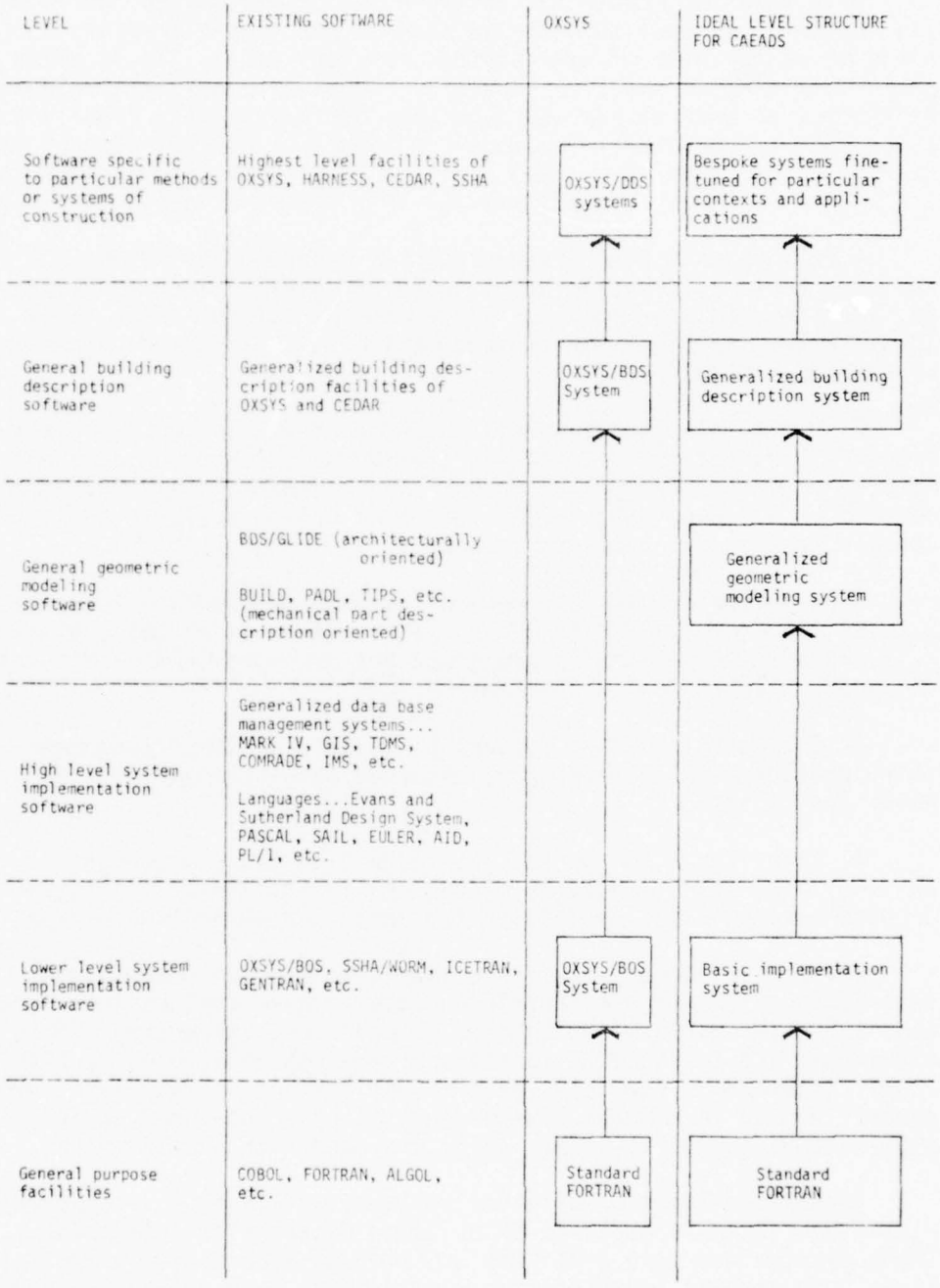


Figure 10. System Implementation Levels

are very precisely adapted to describing buildings within a specific building system or method. Their power is achieved at the expense of severely restricting their range of application.

2.2.5 Levels Required in CAEADS Data Base

The OXSYS system[9] is a clear illustration of the philosophy of implementing a building description data base system in a hierarchy of levels.

The lowest level is standard FORTRAN, because it is widely known, well supported, and very widely implemented. This, in turn, is used to implement BOS, which provides computer-aided design (CAD) system building tools. BOS and FORTRAN are then used to implement OXSYS/BDS, a system which provides very powerful facilities for storing and manipulating descriptions of buildings which generally follow a rectilinear discipline. OXSYS/BDS can be used directly as a building description system or as a high level tool to implement a variety of highly specialized OXSYS/DDS systems. Each of these DDS systems is designed around a specific component-based building system, such as the Oxford Method of Construction. DDS thus exploits the rules of each system to provide extremely powerful capabilities such as automated component selection, sizing, and detailing.

The right-hand column of Figure 10 illustrates one possible five-level structure for the proposed CAEADS data base. At the lowest level --as in OXSYS,HARNESS, CEDAR, and SSHA --is FORTRAN. It is widely available, well supported, and well known by the application programmers available to the Corps and to A/E firms.

The next level could be BOS or some equivalent system. Built onto this should be a generalized geometric modeling system like Eastman's BDS[10] (not to be confused with OXSYS/BDS) or GLIDE. This level is omitted in OXSYS, because OXSYS/BDS deals only with relatively restricted geometries. However, it certainly would be needed in a system which, like CAEADS, is intended to deal with a wide variety of different building geometries and methods of construction.

The generalized building description system corresponds in level to OXSYS/BDS, but does not assume any particular geometry or method of construction. It should both function in itself as a

---

9 E.M. Hoskins, Integrated Computer-Aided Building and the OXSYS Project (Applied Research of Cambridge Ltd., June 1976).

10 C.M. Eastman, Preliminary User's Manual for BDS 10, Institute of Physical Planning (Carnegie-Mellon University, September 1976).

building description facility and also facilitate simple and rapid implementation of a wide variety of specialized systems in different District and A/E offices. At this top level, systems would be oriented towards different methods of construction and different building types, as required. This diversity of specialized systems at the highest level is extremely important, because attempting to develop a single system suitable for use by all the design groups and all the very diverse building tasks carried out by the Corps would be unworkable and undesirable.

2.3 CONTENTS OF THE PROPOSED CAEADS DATA BASE

2.3.1 The Logic of Building Description

A building can be described, first of all, as an assembly of functional volumes such as rooms, zones, and departments. These functional volumes each have a geometry, a location, and various nongeometric properties such as occupancy, lighting level, or ambient temperature. Some of these functional volumes have unusual or unique properties, some are instances of standard types, and some represent relatively minor modifications of standard types.

Secondly, a building can be seen as an assembly of physical components: columns, beams, slabs, ducts, pipes, etc. Each of these components also has a geometry, a location, and nongeometric properties; they may also be unique objects, instances of standard types, or modifications of standard types.

Thus, at the lowest level, a building design can be represented as a set of elements with their associated properties, and the building design process can be thought of as selecting or creating elements, assigning properties to elements, and assembling elements into geometrically and functionally related systems.

These elements can be assembled into entities called subsystems; for example, a set of functionally interrelated rooms may form a department, a set of connected structural members may form a frame, or a set of pipes and fixtures may form a drainage network. These subsystems will have global properties which may be of interest, such as the overall shape and area of a department. Furthermore, just as there are geometric and functional relations between elements, there may be geometric and physical relations between subsystems. Subsystems in turn can be assembled together to form higher level subsystems, and so on, as shown in Figure 11. Since many elements, details, and even subsystems are repeated throughout a building, it is customary to factor a building description to eliminate as much redundancy as possible.

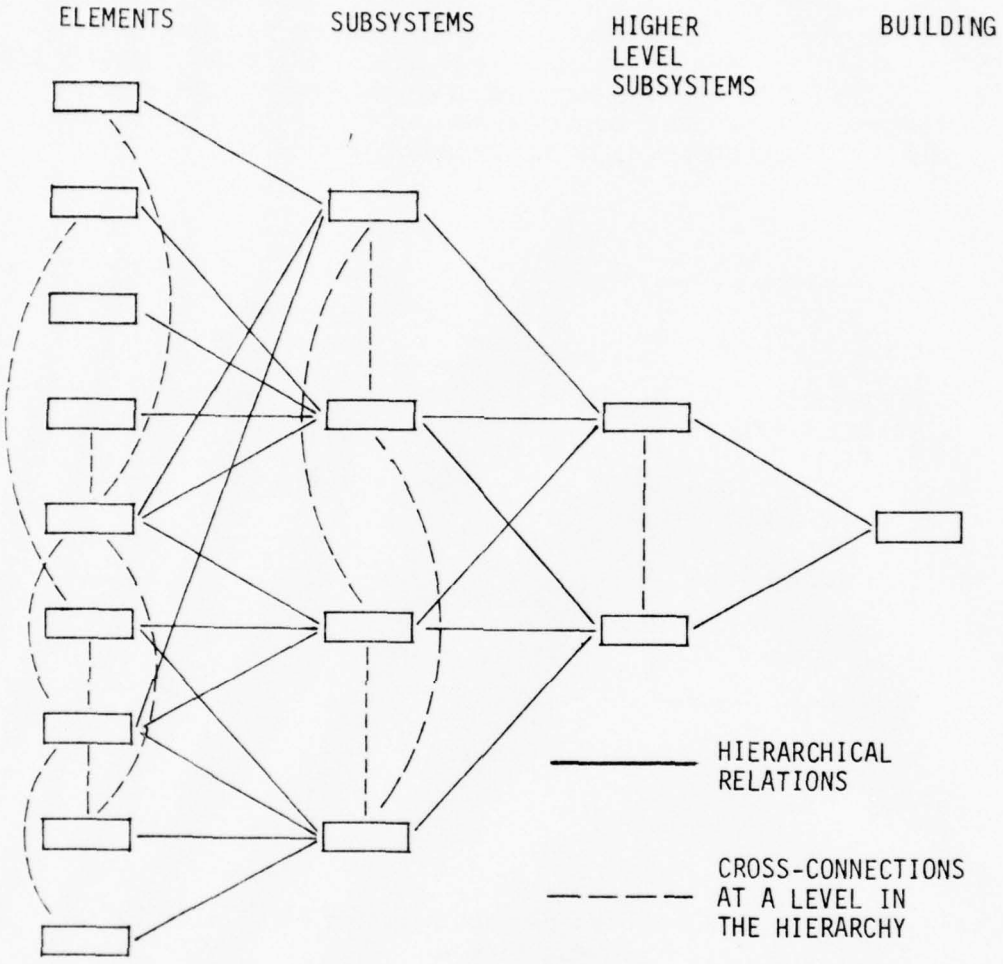


Figure 11. Hierarchy of elements and subsystems in a building.

A traditional set of working drawings locates and identifies elements in the building; the drawings are cross-referenced to schedules, specification clauses, and detail sheets which contain detailed descriptions of particular components, materials, and details. A similar strategy is followed in computer-based building description. A project file contains records which locate and identify elements; this file is cross-referenced to catalogue files which contain detailed element descriptions, specification clauses, etc. Figure 12 illustrates this arrangement schematically.

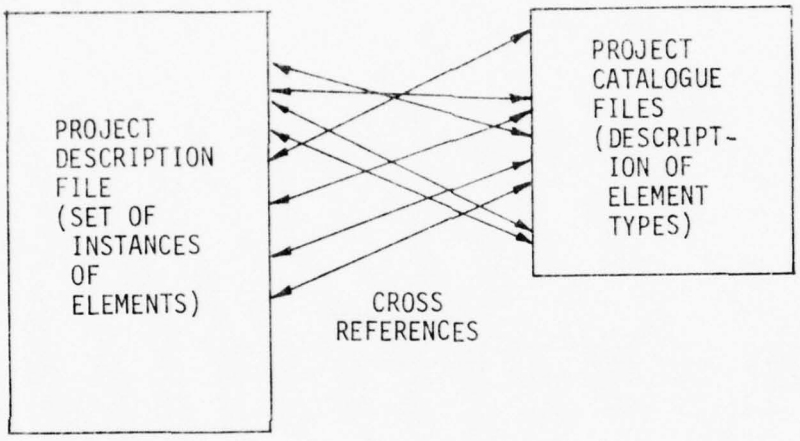


Figure 12. Relation of the project description file to the project catalogue file.

The data in the project catalogue files will mostly be a subset of data contained in larger general reference catalogues, just as the data contained in a normal project specification are mostly a subset of data contained in some master, or guide, specification. This arrangement is schematically illustrated in Figure 13.

Computer-aided design systems intended for use with a relatively restricted and well-defined building system or method are often able

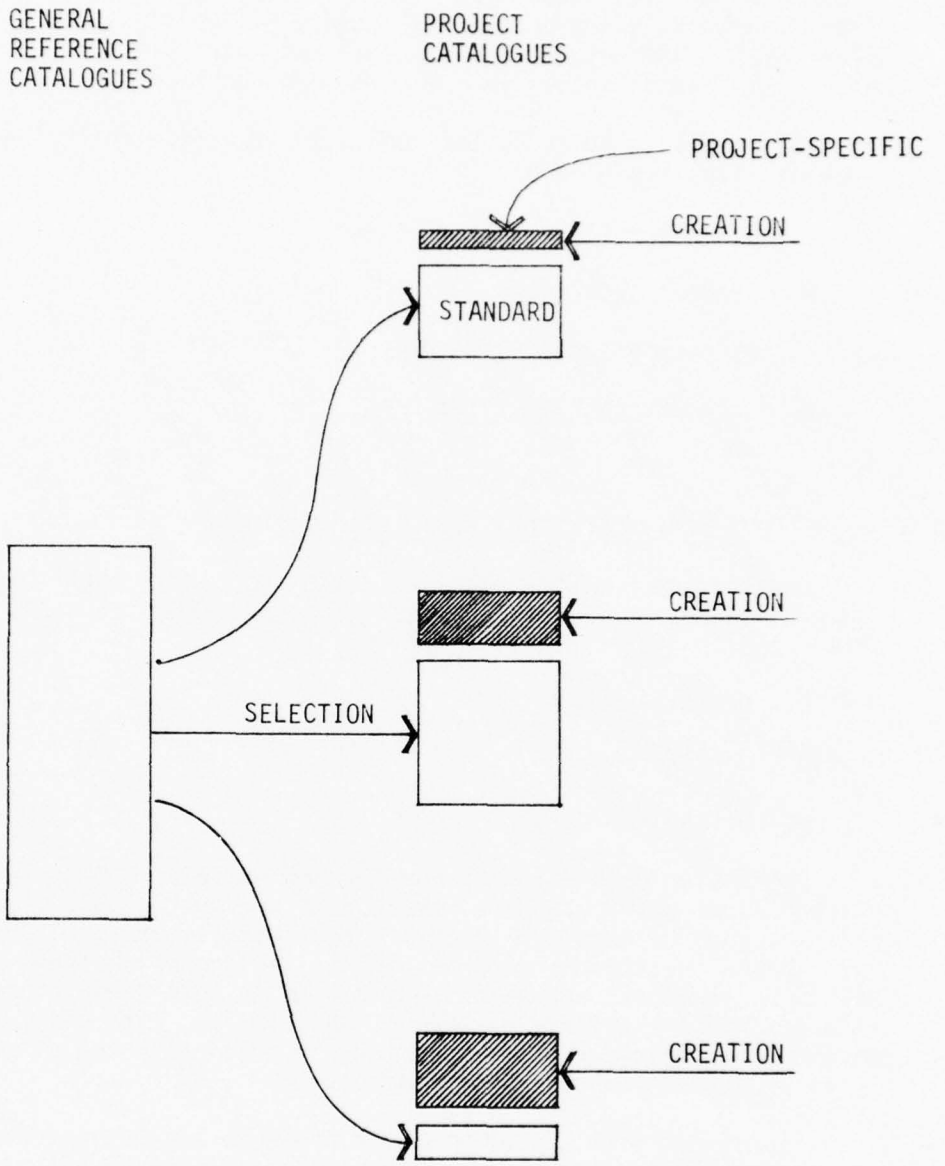


Figure 13. Creation of project catalogues from general reference catalogues.

to dispense with the distinction between project and general reference catalogues, since a complete catalogue for the system may be quite small. However, this is not the case with a system like CAEADS, which is intended for use in a more general context.

This analysis leads to the conclusion that the following basic types of files are needed:

a. General reference catalogues

b. Project catalogues

c. Project description files

d. Site description files.

2.3.2 Definitive, Working, and Historical Data

Consideration of the way the design and construction process flows through time leads to a second classification of data; the categories in this classification are:

a. Definitive data

b. Working data

c. Historical data.

Definitive data describe the current state of the design. In a traditional design process, these data are in the form of a master reference set of drawings which are periodically updated. Working data in the traditional mode consist of designers' sketches and notes. Ideas are developed and experimented with in this form before being entered into the definitive project description. The point at which working data become definitive data is a critical point at which the chief project designer exerts control.

Experience with use of OXSYS on projects has demonstrated that this basic distinction between definitive and working data is effective in practice. However, it has also shown that the idea of a project having a single definitive type of data is a little too facile. In practice it is common for several contending alternatives to be developed in considerable detail before one is finally chosen. Thus, it is more accurate to speak of a definitive description of an alternative rather than of the project.

After a project is completed, it is important to preserve a

building description as historical data. These historical data may be useful in designing subsequent alterations and additions, for reuse in subsequent projects of a similar nature, or for input to a facilities management system.

The combination of these two modes of data classification yields the range of types of files shown in Table 4.

Table 4

Types of Files

	DEFINITIVE	WORKING	HISTORICAL
General reference catalogue	Master catalogue files	---	---
Project catalogue	Definitive project catalogues	Working project catalogues	Historical project catalogues
Project description	Definitive project description	Working project description	Historical project description
Site description	Definitive site description	Working site description	Historical site description

2.3.3 Catalogue Data

Certain types of catalogue data are universally used in building design. In particular, a material and part catalogue such as Sweets[11] is essential. Libraries of standard plans, details, and master specifications are used by most large architectural offices. The Corps, in addition, maintains large libraries of design guides and standards.

[11] McGraw-Hill Information Systems Company Sweets Catalog Vols 1-3 (McGraw-Hill 1976)

Mitchell[12] discusses a variety of different types of catalogues and their handling by computer in some detail.

Although the issue involves questions of computer hardware and general Corps policy that are beyond the scope of this report, it seems reasonable at this stage to recommend that catalogue data be handled centrally in CAEADS as shown in Figure 14. Large reference catalogues would be stored on a time-sharing machine at a central location, with maintenance and updating being the responsibility of a specially trained staff. This method is efficient and allows

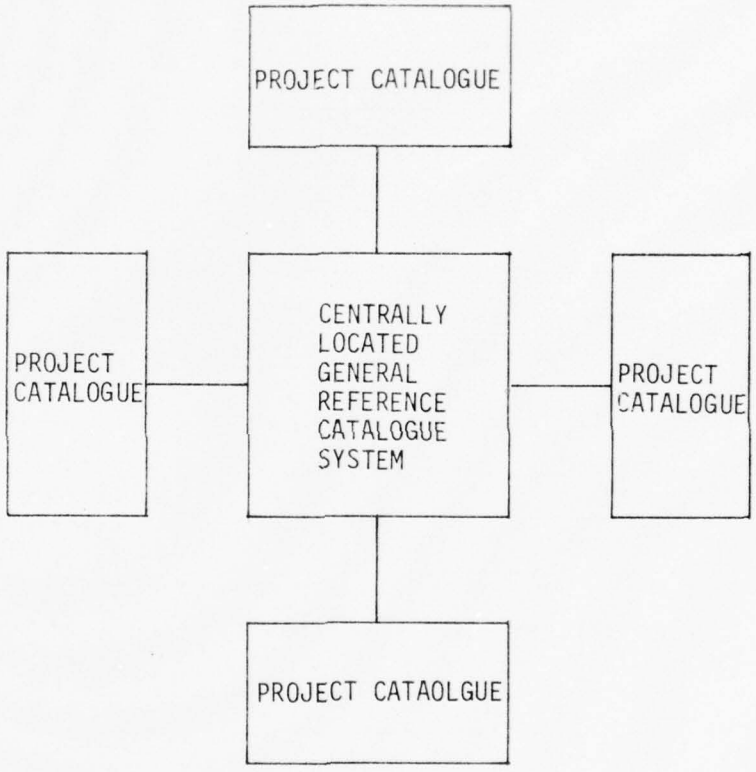


Figure 14. Schematic structure of catalogue sytstem.

close control of quality, integrity, and consistency. Associated with each project (and probably physically located at minicomputer work stations at decentralized locations) would be project catalogues created as required by project designers from the central reference catalogue system. In addition, the project catalogues would contain special elements for a particular project which never find

[12] W. J. Mitchell, Computer Aided Architectural Design (Petrocelli-Charter 1977)

their way into the reference catalogue.

The various types of catalogues required to support an architectural design process are used at different stages in the design process, as illustrated in Figure 15. At the earliest stages, catalogues of nongeometric data such as accommodation and equipment lists and space standards assist in generation of the building program. At the sketch design stage, geometric objects composed of "empty space," such as standard rooms or zones may be selected from catalogues and assembled to define a scheme. Catalogues of aggregated engineering and cost data and relevant performance standards may also be used at this stage. At the detailed design and documentation stage, material and component catalogues, engineering property and cost catalogues, standard details, master specifications; and libraries of standard drafting symbols are useful.

2.3.4 Project Data

The contents of a definitive project data file describe a building in different ways at different stages in the design process, as shown in Figure 16. At the earliest stages, the description usually is nongeometric, consisting of a list of spaces with associated area and other requirements (i.e., the "program"). Next, some circulation relations between spaces may be defined, giving rise to a "bubble diagram" or "interaction matrix." In more mathematical terms, it can be said that a set of spaces has been given a topology. Next, a general layout at a "concept design" level of detail may be produced (the topology is given a geometry). The geometric elements manipulated at this level are chunks of "empty space," i.e., zones containing rooms of particular shapes and dimensions. Finally, the "detailed design" defining precise geometry and physical details of construction is produced. The geometric elements manipulated at this level are mostly "solid objects" such as columns, beams, slabs, ducts, and fittings. Such a well-defined top-down design strategy may not be followed in every case, but this is generally an accurate picture of the way in which architectural design processes are structured.

From an information-processing point of view, this can be viewed as a process of taking an initial description of a building at a very low level of resolution and detail and with very little structure (in the mathematical rather than the engineering sense), and gradually transforming it into a description at an increasingly higher level of resolution and detail. The essential point to recognize is that this transformation is an extended process of incremental decision making; it does not take place in discrete jumps

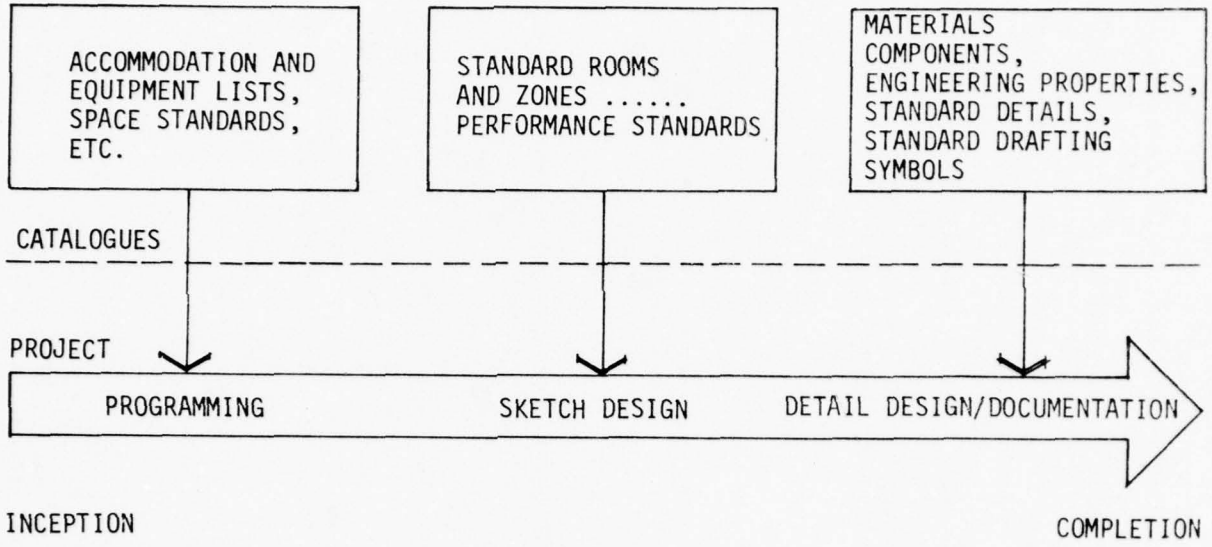


Figure 15. Use of catalogues in building design.

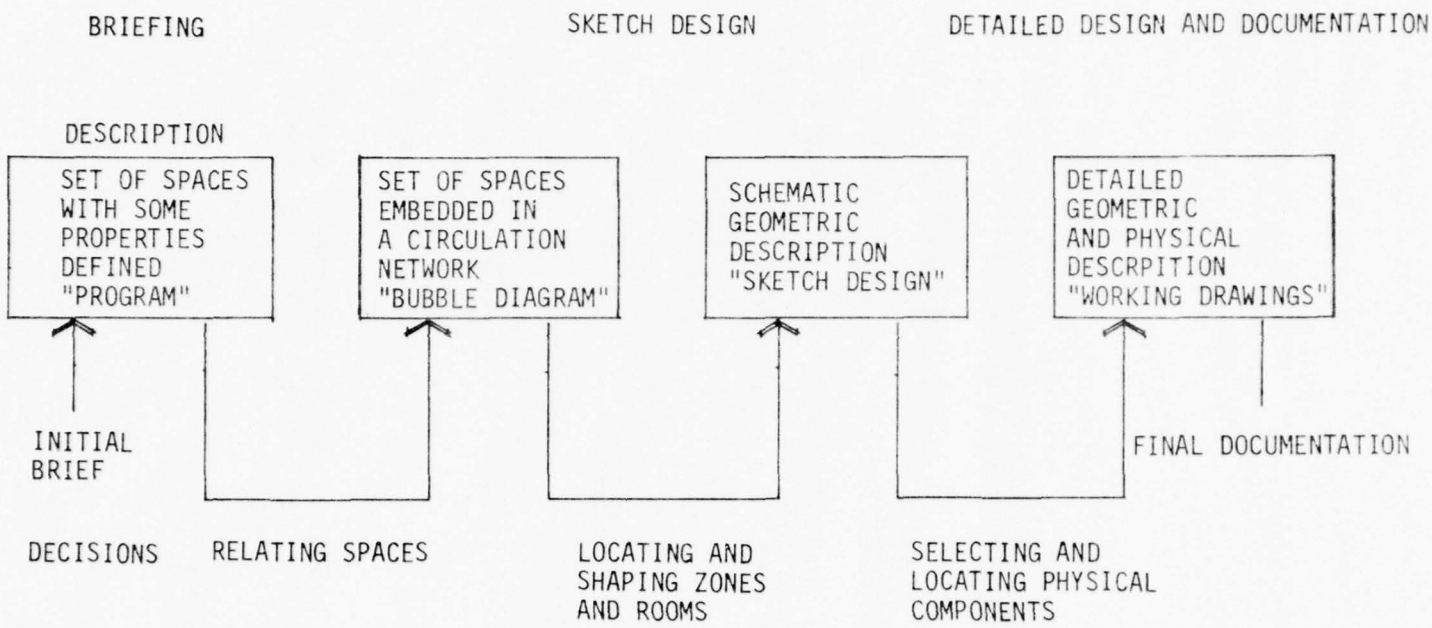


Figure 16. Transformaton of the project description.

from "program" to "concept design" to "working drawings and specifications." If CAEADS is truly to be a design aid and not just a passive data storage system, the philosophy that the definitive project description file is a single, continuously and incrementally transformed entity which develops as the project progresses must be adopted. Both OXSYS and GLIDE, the two most advanced building description systems currently available, exemplify this philosophy.

2.3.5 Site Data

In addition to describing buildings themselves, it is also necessary to describe their sites. Storing site descriptions separately from building descriptions is convenient, since

a. The characters of site and building description data differ

b. The data may become available at different times

c. There is not generally a one-to-one correspondence between buildings and sites (several alternative buildings may be proposed for a single site, or a single design might be executed on several sites).

2.3.6 Size of Project Description

There are generally three interrelated factors which influence the amount of memory needed to store a description of a particular building in a given building description system at a defined level of detail. These are

a. The physical size of the building, as measured in square or cubic feet.

b. The average number of distinct components per unit volume. For example, a highly serviced hospital will have more components per unit volume than a simple warehouse.

c. The average complexity of the individual components, to be represented as measured by the amount of information needed to describe each one.

Very large differences can arise in the sizes of descriptions of the same building in different building description systems. This is because there are large trade-offs to be made between

indexing overhead and the speed with which data can be accessed. An elaborately indexed description will be very much larger than one which has minimal indexing.

The following rough formula may be used to estimate the size of a building description:

$$\text{Words} = C\,(A + I)$$

where C = total number of physical and spatial components represented.
A = average number of words required to describe a component, and
I = average number of words needed to index a component.

Using this formula, building description sizes which may arise under various circumstances can be explored. One may begin by assuming that the geometry of a component is minimally represented by a bounding rectangular parallelepiped. The dimensions and location of a parallelepiped may be represented by nine numbers (three dimensions, three coordinates, and three rotations). An additional number is required as a pointer to the catalogue. Thus, it can reasonably be assumed that a minimum of 10 words/component for geometric description is required.

A simple spatial indexing system could divide a building into cubical cells, and index components spatially by recording the cells which they intersect. If the cells were large by comparison with components, then most components would intersect only one cell. Alternatively, if cells were small by comparison with components, then most components would intersect many cells. It could be conservatively assumed that each component intersects 2 cells. Then the number of words in a geometric description would be given by:

$$\text{Words} = C\,(10 + 2)$$

More elaborate indexing could increase this considerably.

Experience with the BDS system at Carnegie-Mellon University suggests that description of a building as an assemblage of polyhedra (rather than simple rectangular parallelepipeds), with adequate indexing to support real-time manipulation of the design, raises the number of words needed to describe a component to around 40 to 50. It may be estimated that a very elaborate description in a very sophisticated system could require as much as 100 words/component.

Figure 17 plots the total number of words against number of components described for 12 words/component, 50 words/component, and 100 words/component.

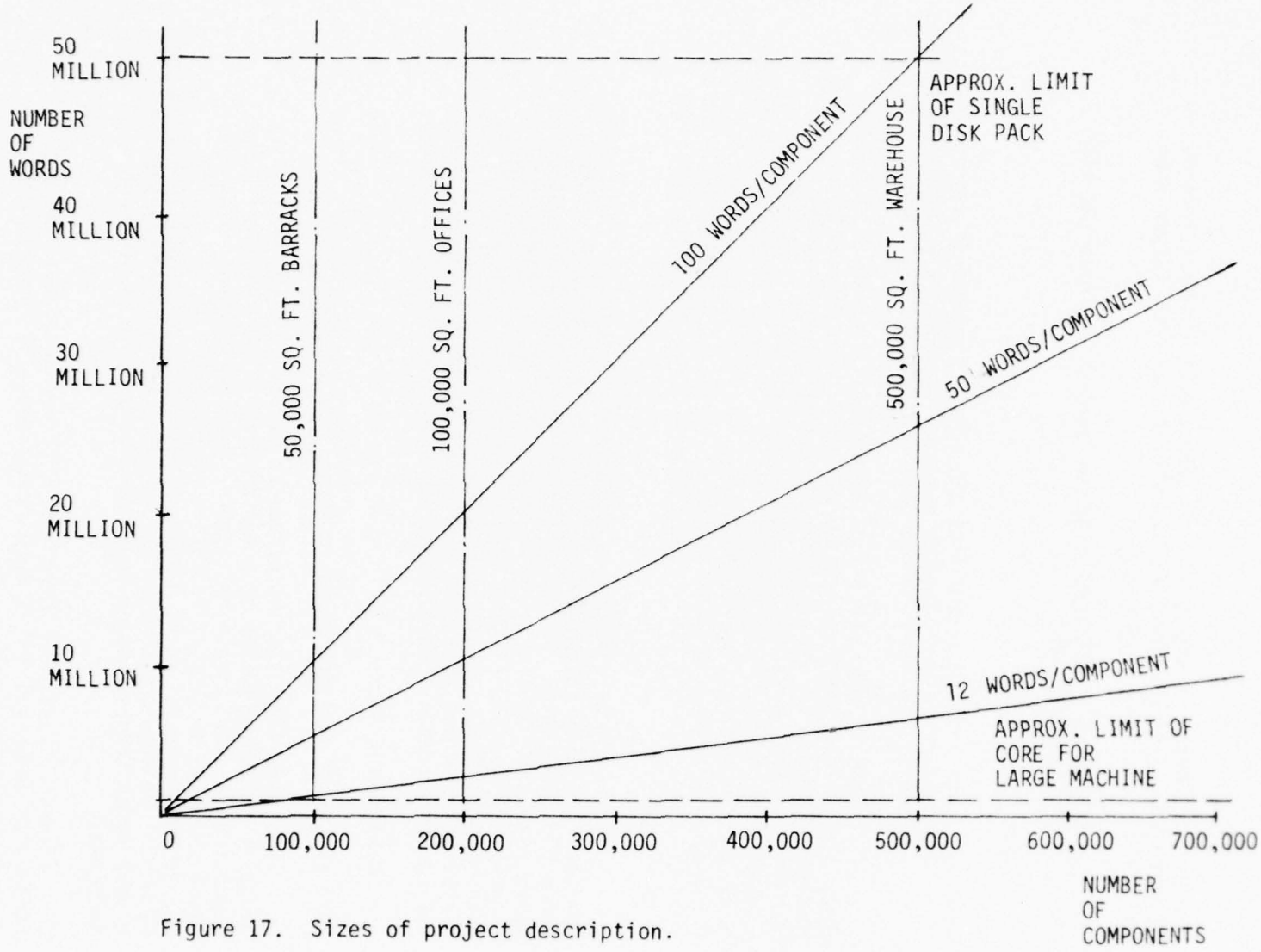


Figure 17. Sizes of project description.

The number of components in a description depends upon both the inherent complexity of the building to be described and the level of detail that is required. A reasonable assumed maximum (based upon experience with OXSYS) would be 10 components/square foot for a detailed description of a very complex building such as a major, heavily serviced hospital. A standard general hospital may require about half this, and a building of medium complexity such as an office somewhat less again. Reasonable rough estimates for the numbers of components in detailed descriptions of representative buildings are shown in Table 5.

Table 5

Rough Estimates of Numbers of Components in Detailed Building Descriptions

Type	Components/sq.ft.	Floor area	Number of components
House	2	2,000	4,000
Barracks	2	50,000	100,000
Offices	2	100,000	200,000
Warehouse	1	500,000	500,000
Hospital	5	200,000	1,000,000
New Walter Reed Hospital	10	1,000,000	10,000,000

These estimates are plotted in Figure 17, together with order-of-magnitude indications of the capacities of various storage media. While all the figures are very approximate, several clear conclusions can be drawn:

a. Only very small buildings or very simple descriptions are likely to be possible with in-core data structures.

b. It can be expected that a very wide range of buildings could be represented in detail within the constraint of fitting each one on a single disk-pack.

c. The largest and most complex buildings may be beyond the capacity of any building description system that it would be reasonable to attempt to implement for CAEADS.

2.3.7 Size of Project Catalogue

In the limiting case, there could in theory be as many components in the project catalogue as there are in the project description. However, in most practical situations the number of different cata-

logue components will only be a small fraction of the number of instances located in the project description.

A description of an instance located in the project description requires in the region of 10 to 100 words, whereas the detailed description of a component in the project catalogue is likely to require hundreds or thousands of words. However, experience with OXSYS suggests that the project description rather than the project catalogue is the dominant factor in overall data base size.

2.3.8 Size of General Reference Catalogue

General reference catalogues have different characteristics than the project descriptions discussed thus far; a comprehensive general reference catalogue can be very large and expensive to maintain. For example, a comprehensive building products catalogue for the United States would probably contain several hundred thousand to half a million entries.[13] Even conservatively assuming 1000 bytes per entry, this implies a data base of around five hundred million bytes. At least several thousand, probably tens of thousand of new entries would need to be coded, checked, and entered each year. An almost comparable number would need to be deleted.

A very large building products catalogue of this type is certainly not necessary for the success of CAEADS, and it may not, in fact, confer benefits commensurate with its cost. However, the example does illustrate the potential magnitude of a general reference catalogue. It is worth noting that the experience of British systems such as OXSYS, HARNESS, CEDAR, and SSHA is not a reliable guide with respect to this issue, since they all assume a systematized approach to building using a relatively limited set of components and details.

To devote a great deal of effort in the early stages of CAEADS to developing large general reference catalogues is unwarranted. It is more important, initially, to implement and put into use an effective project description with minimal general reference support. As this system is used on real projects, it will become clear which expanded general reference catalogue facilities are required, and these can then be implemented step-by-step.

[13] R.P.G. Pennington, "Computerized Product Selection by Performance," Architecture Canada (June 1967, pp 33 - 37).

2.3.9 Supporting Different Disciplines and Applications From a Single Data Base

The essential corollary to the proposition that there should be a single, definitive building description data base is that there should be very powerful, flexible, and convenient facilities for producing subsets of these data formatted in specific graphic, printed, or machine-readable ways, as required. As Klotz's report[14] strongly emphasizes, designers must be able to obtain just the data that they need, in just the form that they need it, with a minimum of effort. An indiscriminate dump of data is useless; it is also useless to expect a designer to write a complicated program to get needed data.

Just as designers need appropriately structured subsets of data, so do application programs. Rather than rewrite existing application programs to conform with the structure of the data base, it is more sensible simply to generate appropriately structured input files from the data base as required. Even when new application programs are to be written, it is likely in many cases to be more efficient for them to operate upon appropriately structured subsets of the data than upon the central data base.

This issue is not unique to computer-aided design data bases; it is also met in business data base applications. It is generally dealt with by providing a report-generation facility which can be employed to produce the required subsets. Report generators for business data bases can usually perform search, select, and sorting operations such as some simple tabulation of quantities, and can output the results either to a file or in printed form in a specific format. Computer-aided architectural design requires some additional types of report-generation facilities: (1) graphic reports in the form of displays and plotted drawings and (2) software to perform data expansion and aggregation operations (this point is clarified in the following subsections on data redundancy and consistency).

Three types of report generators used in computer-aided design systems can be distinguished:

a. Hard coded

b. Hard coded with parameters

c. Very high level language.

14 L. H. Klotz, CAEADS - Critique and Recommendations (February 24, 1977).

The hard coded type is simply a routine permanently "built-in" to the system, which produces a commonly required standard subset of data. This type of routine may be parameterized to obtain some variations (e.g., a drawing expressed with different graphic options). Provision of a very high level language interface allows a designer/user to write very simple, concise, and comprehensible programs to generate types of output or files that are not standard in the system. This type of facility has been very successful in some of the more sophisticated business data base management systems.

Hard coded report-generation routines with or without parameters are commonly used in conjunction with computer-aided design data bases. Use of very high level language interfaces has been less common to date. The CAEADS system will need a wide range of report-generation facilities, probably encompassing all three types.

2.3.10 Reintegrating Data Generated by Application Programs

After an application program has been run, it may be desired to reintegrate a file which it has generated into the building description. This process may be considered to be the converse of report generation. Hard coded data reintegrators could be written, or some kind of generalized facility for defining reintegration operations could perhaps be developed.

However, data reintegration in general is a more complex task than report generation, since data generated by an application program may be inconsistent or incompatible in some way with data already in the data base.

2.3.11 Access Facilities Required

Below the report-generation and data reintegration routines exists a set of basic access routines which operate upon the various files in the data base to retrieve, insert, delete, and alter information. To achieve system flexibility and modularity, it is essential that

a. This set of access routines be well-defined

b. The form of calls to these routines never be altered

c. All operations upon data be through these routines.

As Eastman says, "Thus, if the format or contents of the data is ever changed, only the subroutine calls will need to be altered and not the code throughout the system that is using the data".[15] Figure 18 shows the types of interfaces between users or application programmers and the data base that are created using this approach. This concept can be extended to provide a whole hierarchy of levels of insulation between various levels of software and the data.

An alternative approach, which becomes possible using a CODASYL-style data base management system, is for all application program access to be via the system's data manipulation language (DML) facility. This has the same result of insulating code from the effects of possible changes in the structure or content of the data base.

Regardless of whether access routines or a DML are employed as the access mechanism, appropriate access structures must be provided in support -- namely indexes or similar kinds of mechanisms to facilitate access to the data. In a building description data base, multiple paths of access through the data are generally required. These are discussed in detail in Chapter 3.

2.4 THE PROCEDURAL MODELING CONCEPT

2.4.1 Definition

Data can become available to an application program in two ways: it can be retrieved from memory or it can be computed from other data. Thus, building descriptions can be implemented in two basic ways: either by explicitly storing all the data that is needed for input to application programs, or by storing some minimal subset of that data and providing routines for generating the rest. Most practical computer-aided design systems utilize both stored and generated data. However, a distinction can be drawn between descriptions which mostly represent data in explicit, stored form, and descriptions which rely extensively upon generation of data by programs as needed. The first type of model may be termed a passive data structure, and the second type a procedural model.

There are good reasons to suggest that a strongly procedural approach to building description has considerable advantages over a passive data structure approach. Since the use of a strongly procedural approach has numerous implications for the nature and organization of the data base software, it will be worthwhile to examine the concept in some detail. Analysis of the nature of redundancy in data provides a convenient starting point.

[15] C. Eastman, Assessment of Work on CAEADS (February 2, 1977)

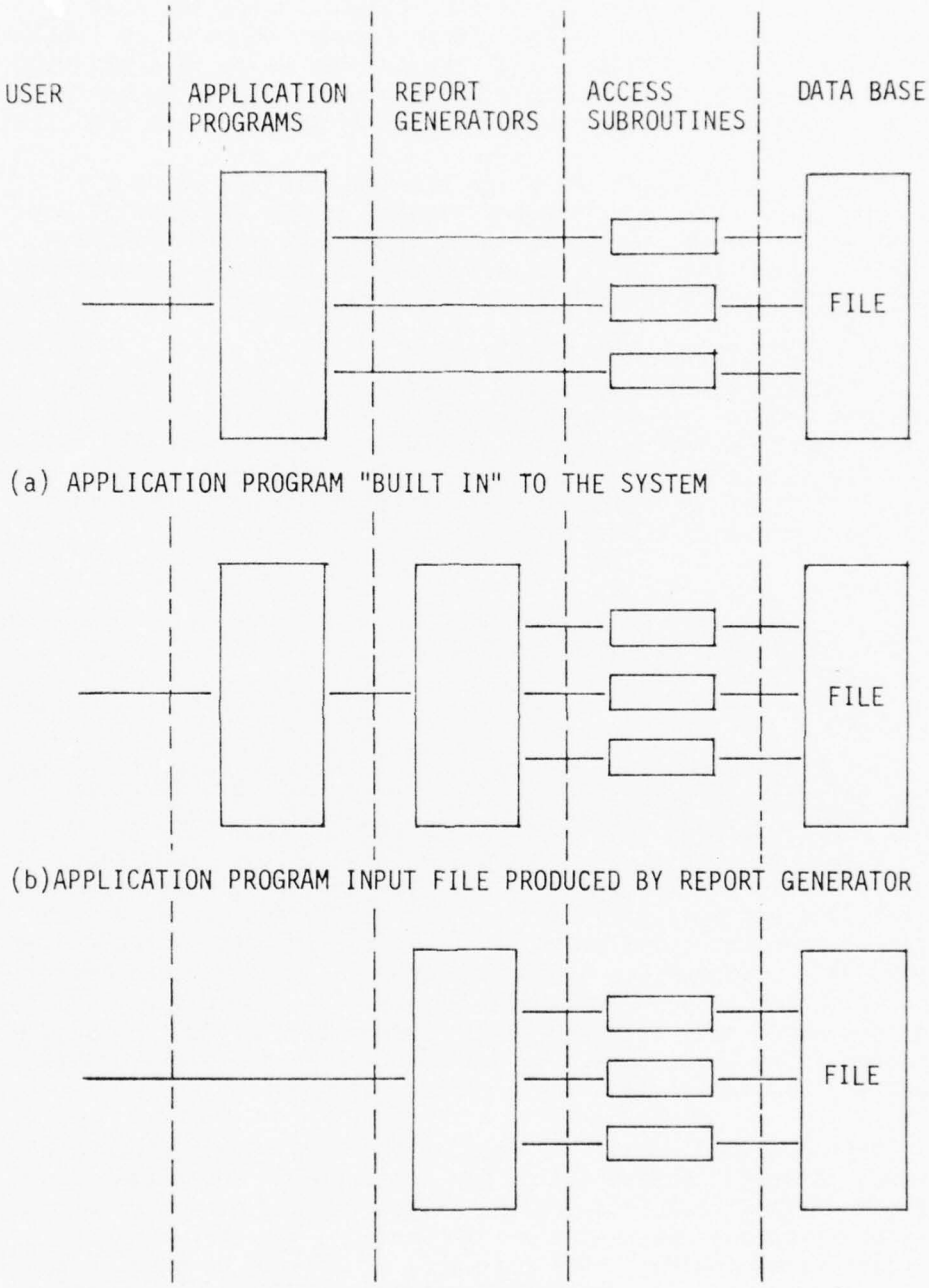


Figure 18. Forms of interface to the data base.

2.4.2 Redundancy

A data base demonstrates redundancy if it contains information that can be computed from other information in it. For example, the representation of a rectangle might be described by coordinates of four vertices, an area, and a perimeter (i.e., ten numbers), as follows:

X_1 , Y_1
X_2 , Y_2
X_3 , Y_3
X_4 , Y_4
AREA
PERIMETER

This is clearly a redundant description, because, for instance, area and perimeter can be computed from the coordinates. A nonredundant kernel of data describing the rectangle might be origin coordinates plus length and width (i.e., four numbers), as follows:

X_0 , Y_0
LENGTH
WIDTH

In order to expand this kernel into the previous redundant description, the following functional relations are required:

$X_1 = X_0$
$X_2 = X_0 + \text{LENGTH}$
$X_3 = X_2$
$X_4 = X_1$
$Y_1 = Y_0$
$Y_2 = Y_1$
$Y_3 = Y_0 + \text{WIDTH}$
$Y_4 = Y_3$
$\text{AREA} = \text{LENGTH} \times \text{WIDTH}$
$\text{PERIMETER} = 2(\text{LENGTH} + \text{WIDTH})$

This set of functional relations may be regarded as a procedure and could be expressed in a programming language like FORTRAN or ALGOL. The variables X_0, Y_0, LENGTH, and WIDTH are the parameters of the procedure, which could be called as follows:

CALL RECTAN (XO, YO, LENGTH, WIDTH)

Execution of the procedure results in procedural expansion of the data; the object (i.e., the rectangle) is then said to be procedurally modeled.[16]

It is important to note that the procedure does not model one particular rectangle, but rather the relations which define an object as being a rectangle. A particular instance of a rectangle is described by the logical intersection of a particular set of parameters with the procedure; a complex object composed of rectangles can be described by the logical intersection of a list of sets of parameters with the procedure.

This approach can be extended by developing a library of procedures to describe a range of different types of primitive objects (e.g., rectangular parallelepipeds, wedges, pyramids, hemispheres, etc.), and to describe complex objects like buildings as assemblages of instances of these primitives. In addition to the basic primitive procedures, parameterized macro-procedures can potentially be created by combining calls to two or more primitive procedures. For example, a rectangular "shed-roof building" form parameterized by length, width, height, and roof pitch could be created by combining "parallelepiped" and "wedge" primitives.

Not every geometric object will have a concise procedural description, however. It can be seen intuitively that regular objects like circles can be concisely described by a procedure, but very irregular or random objects probably cannot. Indeed, the mathematical definition of randomness recently developed by Kolmogorov and Chaitin[17] is in precisely these terms. Stated in a very oversimplified way, it defines an object as relatively regular if it can be encoded by a short procedure, and relatively random if it requires an extensive procedure.

Thus, the usefulness of the procedural approach to geometric modeling depends to some extent on the character of the objects being modeled. Fortunately, buildings appear to be ideal candidates for procedural geometric modeling. They are composed of large numbers of components, most of which can be described as instances of a relatively few basic models. Furthermore, numerous regularities exist at both the component level and the overall geometric organization

[16] M. E. Newell and D. C. Evans, "Modeling by Computer," in J. J. Allan (ed.), CAD Systems (North-Holland Publishing Company,1977).

[17] A. N. Kolmogorov, "Logical Basis for Information Theory and Probability Theory," IEEE Transactions on Information Theory, Vol IT-14, No. 5, (September 1968) pp 662-664

level.

Nongeometric properties can also be modeled procedurally. For example, if the volume and density of an object are known, then its total weight can be computed by means of a simple functional relation.

Procedures can be written to produce different sets of expanded data for different purposes, for example, there may be many possible "expanded" versions of an object.

There are three very important benefits to be derived from use of procedural modeling techniques in building description:

a. Increased efficiency

b. Facilitation of consistency checking

c. Explicit definition of functional relations.

These benefits are discussed in the following subsections.

2.4.3 Efficiency: the Store Versus Compute Question

When an object is described by a nonredundant kernel of data, as discussed above, procedural expansion must be undertaken whenever information is needed for an application or report. The converse strategy is to store data in an expanded form as required for various anticipated applications. As Eastman[18] explains, "A single beam might be represented as: two joint centers with an edge between them, a set of loads and section modulus (for statical analysis); a mass with known conductance, with surfaces of given area (for thermal conductance analysis); and as surfaces of given reflectance and orientation (for lighting design); plus various sections and plans for drafting. Also, the requirements of each analysis require unique information about various relations between elements. Structural loads are split and transferred through joints, defined as a point, and solved in a partial ordering; thermal transfers are passed through common surfaces, with simultaneous or iterative solution."

The advantages of nonredundant storage in terms of efficiency are that

a. The amount of data stored is minimized

---

18 C. M. Eastman, "Databases for Physical System Design: A Survey of U.S. Efforts" CAD 76 Proceedings (IPC Science and Technology Press, 1976).

b. A description is easily changed simply by altering parameters.

c. The data structure within a procedure can be the most efficient for dealing with the particular type of object that it represents. (For example, it would be appropriate to model a solid of revolution in a different way than a rectangular parallelepiped.)

The disadvantage is that the computational cost of the procedural expansion required whenever information is needed may become significant.

Conversely, storage of expanded descriptions eliminates the computational cost of procedural expansion, since required information is always directly available. However, manipulation and updating become cumbersome and expensive, and much more storage is consumed.

Between these extremes are compromises which may represent optimum balances in particular situations. For example, the non-redundant kernel plus expanded data that are used with particular frequency may be stored. Stating a general rule about whether it is better in terms of efficiency to store or compute redundant data is not possible, since the optimum strategy depends on context. However, what is needed is a modeling technique which allows adjustments based on context. A procedural approach to geometric modeling is more powerful than a passive data structure approach in this respect, since in principle it allows any level of redundancy in the data.

2.4.4 Data Consistency

Because functional relations exist between items of data in a building description, items could have values that are inconsistent. For example, the values of LENGTH, WIDTH, and AREA of a rectangle are interrelated as follows:

AREA = LENGTH x WIDTH

Graphically, the relation can be expressed as shown in Figure 19.

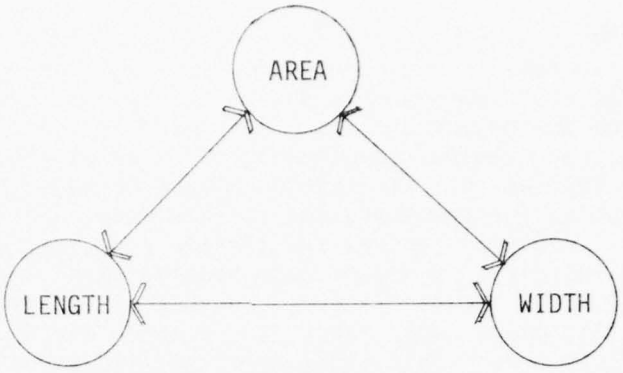


Figure 19. Functional relations.

Specific values for any two of the variables uniquely determine a value for the third. Assigning any other value to the third variable results in a geometrically inconsistent representation.

Another source of possible inconsistency arises if constraints are placed on the values of variables. For example,

AREA = LENGTH x WIDTH
AREA < 100

A set of values for AREA, LENGTH, and WIDTH may be consistent with the geometric relation, but violate the constraint. Constraints are generally thought of as items which are defined at the beginning of a design process and which define a "problem" to be solved, but this can be very misleading. It is more accurate to recognize the design of a building as an incremental process of defining values for a large number of variables, simultaneously building a structure of functional relations between them. Values that are specified early in the process will constrain the specification of later values. Designers following different sequences through a design process will encounter series of "problems" which are constrained in different ways, and which may require very different solution techniques.

The various types of inconsistencies which can arise in a building description due to the existence of functional relations and constraints on the variables do so because a building is a geometric object physically realized in three-dimensional space and simultaneously existing within a particular social and economic context. A design can violate the laws of Euclidean geometry (i.e., be a nonsense object like the paintings of Escher), the laws of physics (i.e., fail for engineering reasons), and the laws of economics (i.e., cost too much), or fail to meet human needs.

The network of functional relations that develops as a building description evolves is very dense and complex. The typical consequence in the later stages of a design process is that altering one aspect of the design can propagate changes widely and in unexpected directions. For example, lengthening the span of a beam alters the loading and implies that the section should be adjusted, with logical consequences for the mass, the surface area, and the appearance of the beam. Consequences may proliferate even further, with frightening rapidity. A deeper beam section might imply a deeper interstitial space between floors; because of building height limitations, this might imply fewer floors,which would in turn reduce the total floor area. This might mean that accommodation requirements could no longer be met.

Functional relations and constraints may or may not be explicitly represented in a building description. If they are not, it becomes the sole responsibility of the designer to ensure that the design contains no inconsistencies. If they are, the computer can be used to check for and report inconsistencies, and perhaps to some extent to automatically correct them. CAEADS' power and usefulness in practice will be directly influenced by the sophistication of the consistency-checking-and-maintenance capabilities; every effort should be made to develop these to the highest possible level.

2.4.5 Consistency Maintenance Software

The topic of consistency maintenance in large, complex data bases is a relatively new one, and little useful theory has been published. However, at least five potential approaches can be described:

a. Linear sequencing of the design process

b. Normalization as proposed by Codd[19]

c. Evaluative approaches

d. Use of operators which always maintain consistency

e. Synthetic approaches based upon procedural descriptions.

Linear sequencing is a very commonly used strategy in computer-aided design systems. This strategy requires that the problem be partitioned into a set of subproblems which must be solved in strict

[19] C. J. Date, An Introduction to Database Systems (Addison-Wesley, 1975) Chapter 6.

sequence. At each stage, certain design variables are fixed and become constraints in the next stage. Mapping programs are used to derive the different variables required as input to the next stage from these known values, by means of known functional relations. For example, it might be specified that design of the structural subsystem of a building must precede the design of the mechanical subsystem. Thus, the span section and mass of a beam would be fixed during the structural subsystem design stage. A mapping program could be written to derive from span and section the functionally related data of surface areas. Only special data such as conductance would be required as extra input to a thermal conductance analysis program.

A technical difficulty arising from this approach is that functional relations tend to be extremely complex and difficult to analyze, making the implementation of mapping programs an exceedingly troublesome task.

An even more serious problem is the necessary acceptance of strictly ordered sequence of stages. (It might perhaps be argued that this could be overcome by developing mapping programs to allow flexibility in the sequencing of stages. However, a combinatorial explosion develops if this is attempted, making the severe limitation of sequencing options inevitable.) In some kinds of design, this may be acceptable; for example, in ship design, the ISDS system[20] accepts quite rigid sequencing. However, in overall building design it is unacceptable. While there is of course a general progression of stages in an architectural design, there is considerable feedback and iteration between stages. Furthermore, different designers have different approaches, and different building types may demand different approaches. A building may be designed "from the outside in," or "from the inside out." In some cases, the structural system is dominant and other systems must be constrained by structural choice; alternatively, space planning considerations may be paramount, or the mechanical system may dominate. There is no general rule. Any architectural computer-aided design system which imposes one individual's conception of an "ideal" design sequence on every project and every user, is doomed to produce poor quality results, and generally to fail. Consequently, it is strongly recommended that the Corps not accept any system design which imposes such a sequence.

The second approach to consistency maintenance -- normalization -- derives from the theory of relational data bases as

---

20 J. Brainin, Functional Description for the Integrated Ship Design System-(ISDS) Report 4663 (Naval Ship Research and Development Center, April 1975).

developed by E. F. Codd and others.[21] The theory is too complex to explain in depth in this report. Briefly, Codd's technique provides a rigorous and systematic method of removing redundancy and describing relations, enabling the updating of any item to be structured to ensure that certain types of consistency relations are maintained. This is undoubtedly a useful approach, but as Baer and Eastman[22] point out, it is far from being a complete theory for dealing with the kinds of consistency-maintenance problems that occur in a computer-aided design data base.

The third approach -- evaluation -- relies upon the use of programs which identify inconsistencies in the data base. These programs may either inform the designer or attempt to resolve the inconsistencies automatically. A pioneering example of an architectural system which implemented the evaluative approach was Nicholas Negroponte's URBAN 5.[23] Consistency evaluation can be carried out as a batch job at intervals in the design process or as a background operation of the system, or the constraints associated with a data item may be reevaluated whenever that item is accessed or altered. Yang and Fenves[24] and Baer and Eastman[25] provide detailed discussions of approaches to consistency evaluation and automated rebinding of values to regain consistency in computer-aided design data bases.

The fourth approach -- use of consistency-maintenance operators -- is powerful but of limited application. The theory is that all operations performed upon a description should be through a well-defined set of operators, the application of which can be shown to always result in a new description which preserves some specific type of consistency relation. This concept can be illustrated by reference to the familiar two-dimensional integer array representation of a floor plan. The plan represented can be changed by assigning an integer to a location in the array. This operation can never result in the generation of a plan that is inconsistent in the sense that rooms intersect or overlap. A less trivial

---

[21] C. J. Date, An Introduction to Database Systems (Addison-Wesley, 1975) Chapter 6.

[22] A. Baer and C. Eastman, The Consistency of Integrated Databases for Computer Aided Design, Institue of Physical Planning Research Report (Carnegie-Mellon University, 1976).

[23] N. Negroponte, The Architecture Machine (MIT Press, 1970).

[24] J. M. Yang and S. J. Fenves, Representation of Information in the Design-Construction Process, Department of Civil Engineering Report No. R74-1 (Carnegie-Mellon University, undated).

[25] Baer and Eastman, The Consistency of Integrated Databases for CAD

example is the set of Euler operators[26] which can be used to manipulate planar polyhedra while ensuring that the result of an operation will retain a planar polyhedral topology.

One limitation on both the consistency evaluation approach and the use of consistency-maintaining operators is that, at times, it may be convenient or even necessary to create a temporarily inconsistent description. For example, since all structural components cannot be located simultaneously, the structure will appear to be "unstable" while it is being defined.

The fifth, and probably generally most promising approach relies upon the concept of procedural modeling, as introduced in section 2.4.1. It assumes that a design is described by the logical combination of nonredundant parameters with a set of procedures. The design is manipulated by changing the values of parameters. Whenever a procedurally expanded (redundant) description of some part of the design is needed, the appropriate procedures are executed to rebind the values of the dependent variables. Consistency checks can be built into this rebinding process; the system might either report a message to the designer if an inconsistency is found, or attempt to resolve it automatically.

The algorithms used to rebind values in a consistent way may involve only simple derivation of algebraic results, such as the computation of area from a length and width. Alternatively, they may involve sufficiently sophisticated problem solving to justify the name "automated design." An example of an algorithm of the latter type is the nonlinear programming algorithm of Mitchell, Steadman, and Liggett,[27] which is used to generate dimensions for an object consistent with a complex set of dimensional constraints.

The rebinding of values may be necessary in several cases; for example,

a. When an individual redundant data item is accessed

b. When a parameter is updated

c. When generating a report or data set for input to an application program is necessary.

Rebinding might be invoked automatically, or it might be under explicit control of the user.

[26] B. G. Baumgart, Winged Edge Polyhedron Representation, Stanford Artificial Intelligence Project Memo AIM-179 (Stanford University, October 1972).

[27] W. J. Mitchell, J. P. Steadman, and R. S. Liggett, "Synthesis and Optimization of Small Rectangular Floor Plans," Environment and Planning B, Vol 3, No. 1 (1976).

2.4.6 The Importance of Making Functional Relations Explicit

Potential gains in efficiency and facilitation of consistency checking are fairly obvious consequences of a procedural approach to building modeling. A less obvious implication is that an effective way of integrating architectural research and design is provided. The nonredundant kernel of data consists of specific information about a particular design, but the encoded functional relations consist of rigorously stated general design knowledge. This knowledge might, for example, be geometric (e.g., the formula for a circle), it might be scientific (e.g., laws of statics used in structural design), or it might be related to the properties of a particular industrialized building system. New knowledge generated through research can be built into a design modeling system in the form of additional functional relations.

In the traditional manual design process, a large number of explicit decisions must be made, since little advantage is taken of functional relations. In computer-aided design, the more functional relations that are built into the system, the fewer the explicit decisions required of a project designer, since large parts of the design description can be derived automatically from a few key decisions. Among existing architectural systems, OXSYS makes very effective use of this principle. A version specially tailored to the Oxford Method of Building (a post beam and panel component system) requires input only of general building geometry. The rules of the Oxford Method are followed to automatically select components, detail joints, and produce a complete building description at a working drawing level of detail.

The use of relational options in generating a project specification from a master specification is another example of how functional relations can be exploited to reduce the number of explicit decisions which a designer must make.

A good building modeling system should release designers from the need to devote large amounts of their time to aspects of building design which form a logical extension of previous design decisions within a predefined context.

2.4.7 Software Implications of a Procedural Modeling Approach

An equation attributed to Douglas T. Ross is

Model = Data + Structure + Algorithms

In most computer-aided design systems, this is interpreted as

Model = (Data + Structure) + Algorithms

In other words, a "passive" data structure, which represents the design, is operated upon by "active" algorithms, which generate desired results. By contrast, the procedural approach to building modeling can be interpreted as

Model = Data + (Structure + Algorithms)

Here, the "passive" data is reduced to a minimal collection of parameters, and the structure of relations between design variables is mostly embedded in algorithms.

The software implication of this change in emphasis is that two types of facilities must be made available to enable users to model building designs:

a. Facilities for entering parameters and storing them in an appropriate data structure.

b. Facilities for defining procedures, i.e., a programming language.

In most cases, these facilities would be employed by different levels of users. An ordinary project designer certainly would not want to deal with a programming language. He/she would use a predefined library of procedures (evoked by user commands), and describe a design by entering parameters to create and assemble specific instances of these procedures. At another level, architects and engineers with programming skills would develop, refine, and extend the procedure library.

2.5 SECURITY AND INTEGRITY

2.5.1 Control of Access

Data may be considered secure if they are properly protected against unauthorized disclosure, alteration, or destruction. As in other large data base systems, data security is of vital concern in building description systems. Among the more important reasons for careful security maintenance in CAEADS are

a. A building description or catalogue may be extremely valuable, representing many man-years of work.

b. Different consulting firms and individuals may be legally and professionally responsible for specific subsets of the data.

c. Certain data may be of sensitive or confidential nature.

The first means of providing data security is to provide an identification and authorization system. A simple user number and password system should be adequate for CAEADS, but a security system that is more difficult to penetrate could be provided if necessary.[28] Users of a data base should be authorized by a responsible individual, presumably a chief project designer in this case. Authorization would involve

a. Assigning the user a number and password

b. Defining to the system what the particular user is authorized to do.

To control access to data, the system must maintain a user profile which contains user names, passwords, etc., and details of what each user is authorized to do to the data. In addition, there must be some facility for associating access constraints with the data. The assignment of access constraints would normally be the responsibility of the person who created the data. When access of data is attempted, the system can then check to determine whether access should be granted or denied.

It would probably be impossibly complicated, and certainly very frustrating to the user, to apply access constraints at the record level in the definitive building description. A more reasonable approach is to exert control over the creation, and subsequent reintegration into the definitive description, of working files. Access controls can be imposed over the particular category of data that a certain class of user can read into a working file. Once a user has a working file, he/she can operate freely upon it. The subsequent reintegration of data from a working file into the definitive building description should be under the direct and explicit control of the chief project designer. This type of system keeps boundaries of professional responsibility and lines of authority clear.

2.5.2 Corruption of Files Due to Errors

Loss, destruction, or unwanted alteration of data can occur due to user errors or hardware or software breakdowns. The detec-

[28] L. J. Hoffman, "Computers and Privacy: A Survey," Computing Surveys, Vol 1, No. 2 (June 1969)

tion of potentially destructive user errors should be accomplished by interface routines, but it cannot be assumed that errors will never slip through. Although software bugs should not be encountered often in system operation, they are always a possibility in a large system. Hardware failures will take place with statistical regularity. Explicit protection must be provided against these eventualities.

Normally, several different types of utilities are provided in a data base system to provide this protection;[29] for example

a. Detection routines to identify situations which may require steps to protect data

b. Journaling routines for recording transactions on the data base

c. Dump routines to create backup copies of the data base as requested

d. Recovery routines to restore computer portions of the data base from a backup copy and the system journal.

The CAEADS data base system will need to incorporate these types of facilities.

2.5.3 Data Verification

Since data recording and key-stroke errors can always be expected when data are entered into a data base, routines for data verification are a necessary part of a data base system. They assume particular importance with geometric description data, since errors can give rise to freakish topological effects and scale distortions, the source of which may not be intuitively obvious.

A common approach to geometric data verification is to run a series of batch verification programs. For example, Falcon Research and Development[30] reportedly has implemented programs which check data sequence, enclosure of volumes, interference of components, and ordering of surfaces in geometric models.

Alternatively, when a geometric model is created interactively, checks of various kinds can be applied to each item of data or entry.

[29] C. J. Date, An Introduction to Database Systems (Addison-Wesley, 1975) Chapter 20.

[30] T. J. Byrne and J. P. Thompson, Computer Representation of Three-Dimensional Structures, (December 3, 1977)

This spreads the verification effort, rather than concentrating it as a single task.

When an error is found, it may simply be reported, or an attempt may be made to correct it automatically. For example, the site description system employed by the Scottish Special Housing Association[31] incorporates extensive facilities for correcting misalignments of building and paving boundaries introduced by small digitization errors.

2.5.4 Data Sharing Problems

Since CAEADS is intended for use by design teams, problems can potentially arise in connection with shared access to the data base; for example:

a. If two users simultaneously attempt to update an item, a deadlock can result

b. Data may be updated by one user while they are being operated upon for some purpose by another user.

To avoid these situations, some facility must be provided to allow a user to temporarily claim exclusive control over all or some of the data.

This is an issue which can cause considerable practical problems in scheduling work on a CAD system, since extensive updates may take minutes or even hours, and other users are denied access to the data base during that time.

[31] A. Bijl et al., SSHA-DOE Site Layout Project," Phase 1 Final Report, CAAD Studies (Edinburgh University, January 1974).

3 THE GEOMETRIC MODEL

3.1 THE INTERNAL MODEL STORED IN THE DATA BASE

3.1.1 The Distinction Between Geometric Descriptions and Digitized Drawings of Buildings

Since the development of the first computer graphic output devices, numerous systems for providing and manipulating graphic output have been developed. Many of these are suitable for handling plans, elevations, sections, and perspectives of buildings, and some have been used as architectural drafting systems. However, it should be emphasized that a typical drafting system is not a geometric description system and is quite inadequate for the purpose of supporting CAEADS. The inadequacies of drafting systems arise for three reasons:

a. The building descriptions which they generate are geometrically incomplete

b. The descriptions generated are highly redundant

c. The geometric information is inadequately structured.

The geometric completeness of a description may be defined as follows. A complete geometric description of an arbitrary point allows unambiguous determination of whether that point is on any specified line or face, or within any specified closed region. Drawings are invariably geometrically incomplete because they represent a three-dimensional object as a series of two-dimensional projections. Even the most complete set of projections of any but the simplest object is likely to contain numerous ambiguities which can only be resolved by the designer's intelligence and experience.[32] Computer resolution of these ambiguities is an interesting artificial intelligence research issue, but ambiguity should certainly not be gratuitously built into a computer's internal model of a design in a practical computer-aided design system.

It is in the nature of orthographic projections as standardly used in architecture that points may appear in different projections. For example, the X and Y coordinates of a point may be defined in one drawing and the Y and Z coordinates in another. Thus, the Y coordinate is represented redundantly (see Chapter 2 regarding the need to avoid redundancy of data).

It is an axiom of computer science that data to be operated upon

[32] G. Lafue, Recognition of Three-Dimensional Objects from Orthographic Views, Institute of Physical Planning Research Report (Carnegie-Mellon University, 1976).

must be structured in appropriate ways if algorithms are to be expressed concisely and clearly and executed efficiently. A drawing stored as a set of projected coordinates and lines is not an appropriate structure for performance of many operations that are important in architectural computer-aided design. This point will probably only become clear when the details of geometric modeling have been discussed, but an intuitive understanding of the difficulty can be gained by considering the kind of algorithm that would be needed to compute, say, the weight of a complex geometric object given several orthographic projections.

By contrast, a true three-dimensional geometric description system stores representations that are geometrically complete, relatively nonredundant, and appropriately structured. Furthermore, provision of an appropriate graphics interface allows drawings to be generated with any specific scale and projection, from any specific direction, sectioned in any specified way. Thus, a three-dimensional geometric description system is much more general and powerful than a drafting system. It can do all that a drafting system can do and much more besides.

3.1.2 Geometric Entities

The starting point for consideration of how a three-dimensional model of a building might be constructed is to establish some fundamental geometric definitions. The basic geometric elements out of which a three-dimensional geometric object is composed are (Figure 20)

a. Points, which may be encoded as pairs of coordinates in two-dimensional space, as triples in three-dimensional space, or as four homogeneous coordinates[33]

b. Lines, which may be straight, singly curved, or doubly curved

c. Surfaces, which may be planar, singly curved, or doubly curved

d. Volumes, which enclose three-dimensional space.

These basic elements may be combined to form various classes of more complex objects (Figure 21):

a. Networks, which are composed of points and lines

[33] W.M. Newman and R. F. Sproull, Principles of Interactive Computer Graphics (McGraw-Hill, 1973).

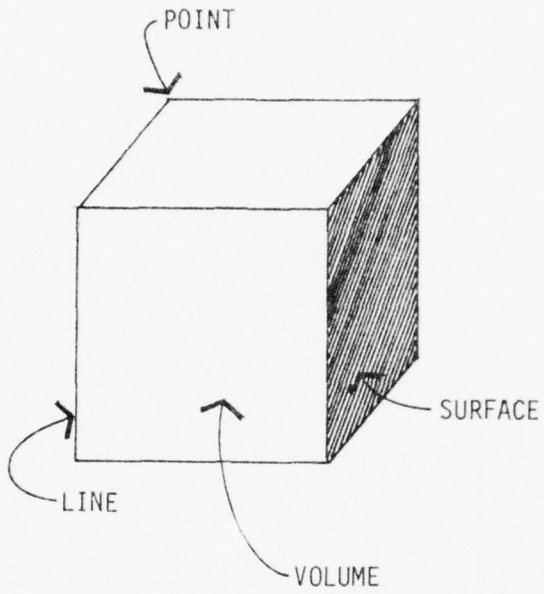


Figure 20. Basic geometric elements.

b. Faces, which are closed parts of surfaces bounded by lines

c. Shells, which are assemblies of faces which do not necessarily bound a volume

d. Polyhedra, in which faces do bound a volume

e. Assemblies of volumes, which are three-dimensional assemblages of solid objects, decomposable into elementary objects

f. Complex assemblies, which may be composed of any or all of the above entities.

A completely general geometric description system must provide facilities for defining entities of any one of these ten types, and for assigning an arbitrary number of nongeometric properties to any one (e.g., the stiffness of a joint represented by a point, the resistance of a wire represented by a line, or the color of a surface).

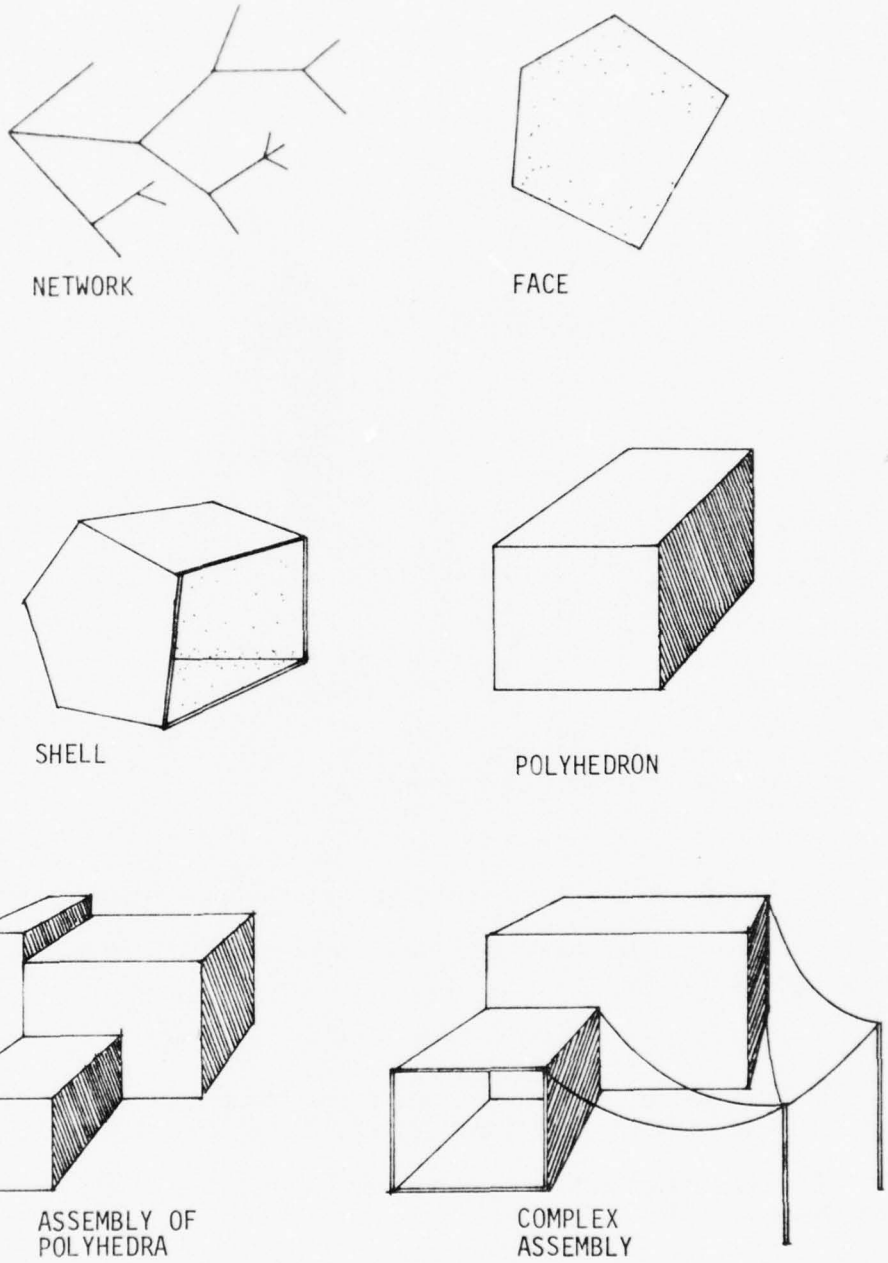


Figure 21. Types of assemblies of basic elements.

3.1.3 Topology

An assembly of geometric entities has a topological structure; a second requirement for a general geometric description system is that it should be able to explicitly represent this topological structure to any required level of detail.

In a network, the topological structure is the pattern of incidence of lines on vertices. This can be numerically represented either as an adjacency matrix or an incidence matrix, either of which can be stored in some compressed form.[34] A floor plan may be regarded as a planar embedding of a network, in which closed regions represent rooms (Figure 22). The dual of this graph represents all adjacencies between rooms, which may be encoded in the same way. Floor plan graphs and their duals are employed in many architectural applications.[35]

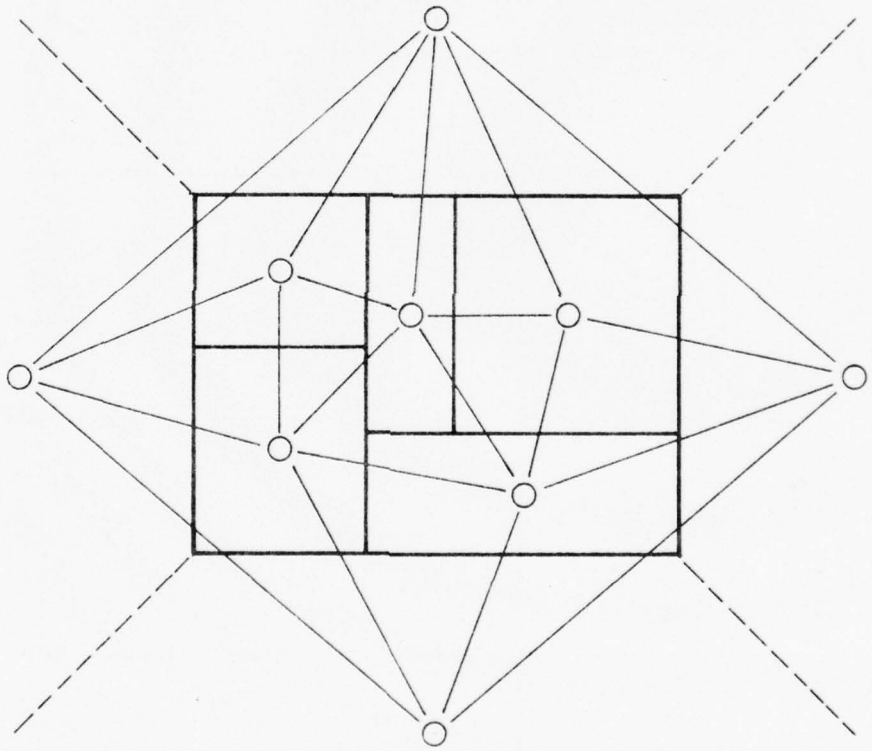


Figure 22. A floor plan and its dual.

[34] W. J. Mitchell, Computer Aided Architectural Design (Petrocelli-Charter, 1977).
[35] Mitchell, Computer Aided Architectural Design

Nine different types of topological relations in a polyhedron may be of interest (Figure 23):

a. The faces surrounding a face

b. The vertices surrounding a face

c. The edges surrounding a face

d. The faces surrounding a vertex

e. The vertices surrounding a vertex

f. The edges incident on a vertex

g. The faces divided by an edge

h. The vertices connected by an edge

i. The edges incident to an edge.

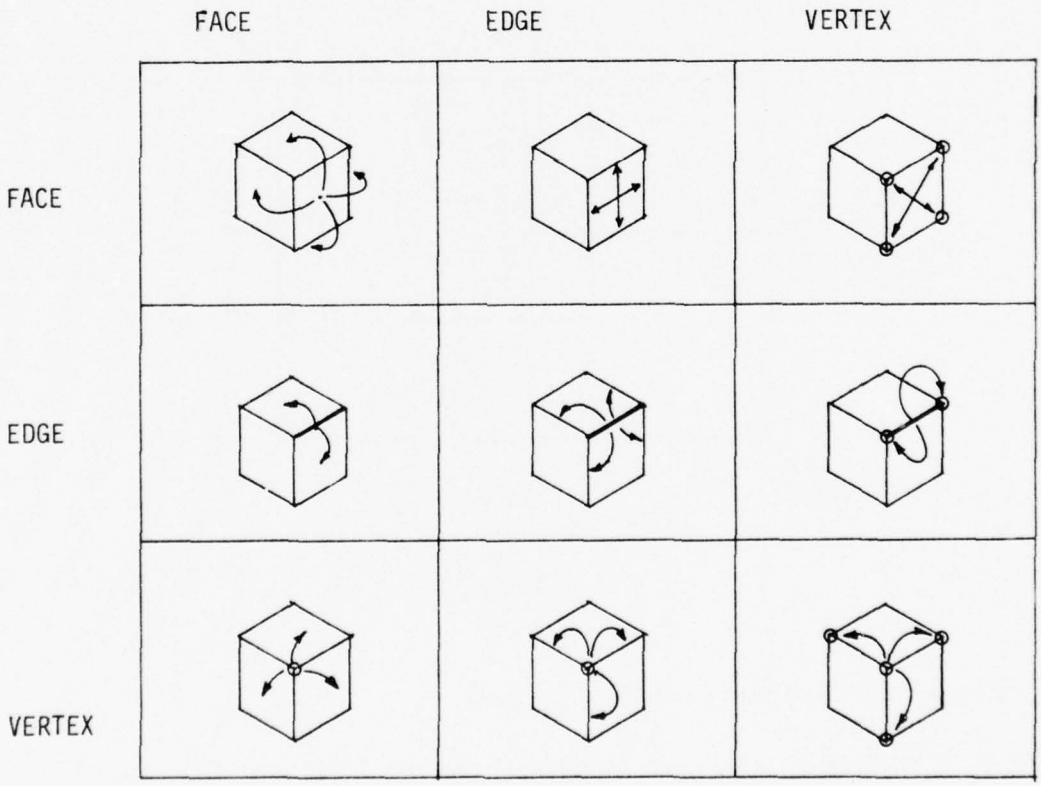


Figure 23. Adjacencies of faces, edges, and vertices.

At least one three-dimensional shape representation method, Baumgart's winged-edge polyhedron method,[36] explicitly represents all nine types of relation. Baumgart's approach has strongly influenced many of the more recently developed shape description systems.

Topological data in explicit form are needed for most engineering analysis applications. For example, for a structural analysis, how a frame is connected must be known; and thermal and circulation analyses are treated as network flow problems. Graphics production usually requires fewer topological data. The most data are needed by hidden line perspective programs, which must know how faces are bounded by lines. Since there are still big differences in the amount and type of topological data needed, shape description techniques oriented towards perspective production applications tend to be inadequate for architectural engineering analysis applications.

3.1.4 Geometry

A topological description records relations between entities, but not shapes or dimensions. Including shape and dimensional data yields a complete geometric description. There are three basic ways of encoding geometric data:

a. *Point set techniques*

b. Boolean techniques

c. Boundary techniques.

The point set technique is the most straightforward. It derives directly from the classical definition of a solid body as a set of contiguous points in Euclidean space. To any specified level of resolution, a region of Euclidean space can be represented by a three-dimensional array in which each location corresponds to a point. A solid object within this space can be described by assigning the value .TRUE. to each point within the object and the value .FALSE. to each point not within the object (Figure 24). More elaborately, a three-valued logic can be used, with the third value employed to represent points on the object's surface.[37] An assembly of many solid objects can be encoded by using a different integer to represent each different solid. This type of technique has been very widely employed to represent building floor plans.[38]

---

[36] B. G. Baumgart, Winged Edge Polyhedron Representation, Stanford Artificial Intelligence Memo AIM-179 (Stanford University, October 1972).

[37] I. C. Braid, "The Synthesis of Solids Bounded by Many Faces," Communications of the ACM, Vol 18, No. 4 (April 1975).

[38] C. Eastman, "Representations for Space Planning," Communications of the ACM, Vol 13, No. 4 (April 1970).

FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE
FALSE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
FALSE	TRUE	TRUE	TRUE	FALSE	FALSE	FALSE
FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
FALSE	TRUE	TRUE	TRUE	TRUE	TRUE	FALSE
FALSE	FALSE	FALSE	FALSE	FALSE	FALSE	FALSE

Figure 24. Point set representation of a shape.

Apart from its straightforward and intuitively appealing character, the point set approach has the advantage of providing spatial indexing. However, it is prodigal in its use of storage where high resolution description of a complex object is required, and its atomistic mode of description is very inconvenient for many applications. These disadvantages make the point set approach impractical for most three-dimensional shape description purposes.

The Boolean approach uses the idea of directed surfaces.[39] A directed surface divides the universe into two disjointed point sets. Points in one set (on one side of the surface) are labeled .TRUE. and points in the other set are labeled .FALSE. A number of different directed surfaces can be defined, and solid objects can be described by performing operations of union, intersection, and difference on point sets (Figure 25). Solid objects thus created

[39] I. C. Braid, Six Systems for Shape Design and Representation - A Review, University of Cambridge Computer Aided Design Group Document No 87 (May 1975)

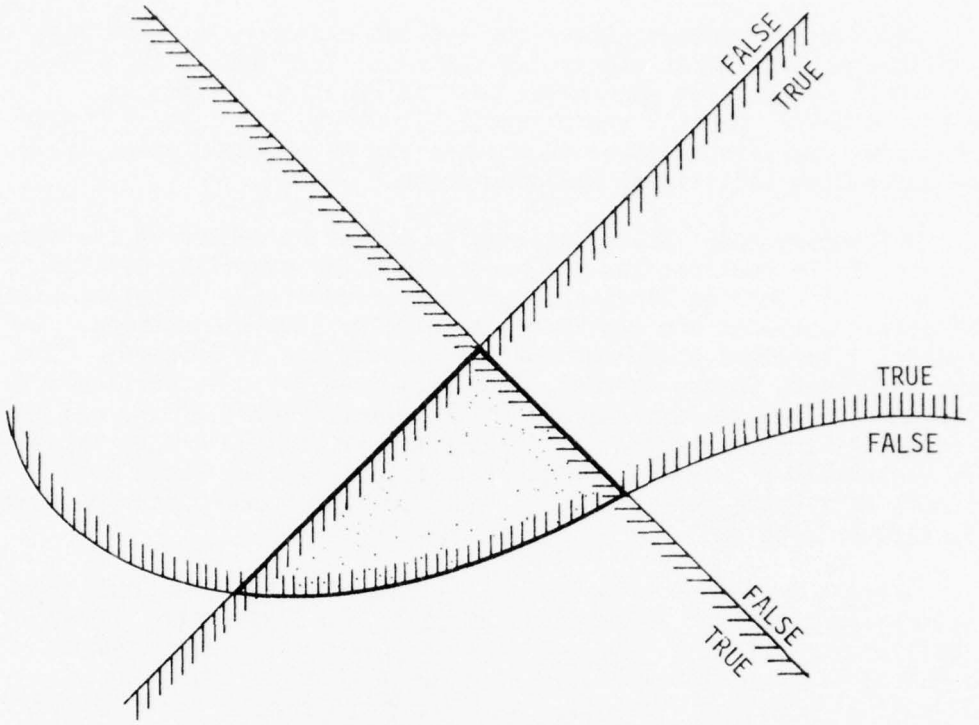


Figure 25. Boolean description of a shape.

can be further combined in various ways using the same Boolean operations.

The Boolean approach is much more economical in use of storage than the point set approach, since it does not require explicit storage of the values of points. It suffices to define the equations for the surfaces, the senses of their normals, and the Boolean operations. However, Braid has identified a key disadvantage: "The disadvantage of a Boolean model is apparent as soon as an attempt is made to draw a component so described. For example, consider drawing a polyhedron of n faces which is held as the intersection of n directed planes. Each of the n planes must be compared with each of the n-1 other planes, the line of intersection found (assuming no planes are parallel), and each line of intersection compared with the n-2 remaining planes to find the portion of the line, if any, lying within the n-2 planes. It is an edge of the polyhedron and can be drawn. To draw all edges requires computation of order n^3."[40]

[40] Braid, Six Systems for Shape Design and Representation - A Review, University of Cambridge Computer Aided Design Group Document No 87 (May 1975)

Despite this disadvantage, the Boolean approach to shape description has been quite widely implemented. The TIPS,[41] SHAPES,[42] and PADL[43] systems for mechanical part description all employ Boolean models. March[44] and Mitchell, Steadman, and Liggett[45] have shown how simplified Boolean techniques can be employed effectively for describing rectilinear building forms.

A boundary model describes a solid object by recording the coordinates of its vertices and the coefficients or equations defining the forms of bounding lines and surfaces (Figure 26). Straight lines and planar surfaces are easily represented by simple equations. The research literature on curved surface description is enormous. The work of Bézier, Coons, Forrest, Riesenfeld and Levin is particularly important. A convenient summary of the current state of the art is provided by a recent set of conference proceedings edited by Barnhill and Riesenfeld.[46] The mathematical theory in general seems sufficiently well developed to deal with any practical architectural form description problem.

A great advantage of the boundary model is that it allows topological and geometric information to be stored separately. Connections between faces, edges, and vertices,which form one component of the model, are stored in some form of incidence or adjacency matrix (see previous section). Real numbers defining vertex coordinates and coefficients in equations are stored elsewhere. This is very convenient, since some shape manipulation operations alter just the topology, some alter coordinates or coefficients, and some alter both. Furthermore, it provides an important opportunity to factor the data, since numerous geometrically differing objects may be instances of the same topological structure (e.g., a cube, a rectangular parallelepiped, and a trapezoidal prism).

Since vertices, edges, and faces of a solid are defined in terms of each other, conversions from one to the other can be made as shown in Figure 27. Thus, it is redundant to describe all three types of entities explicitly (although it may be convenient for some purposes).

41 N. Okino et al., "TIPS-1," Prolamat 73 Proceedings (Budapest, 1973).
42 J. H. Laning et al., SHAPES User's Manual (Draper Laboratory, MIT, 1973).
43 H. B. Voelcker et al., An Introduction to PADL (Production Automation Project, University of Rochester, 1974).
44 L. March, "Boolean Description of a Class of Built Forms," in L. March (ed.), The Architecture of Form (Cambridge University Press, 1976).
45 W. J. Mitchell, J. P. Steadman, and R. S. Liggett, "Synthesis and Optimization of Small Rectangular Floor Plans," Environment and Planning B, Vol 3, No. 1 (1976).
46 R. E. Barnhill and R. F. Riesenfeld, Computer Aided Geometric Design (Academic Press, 1974).

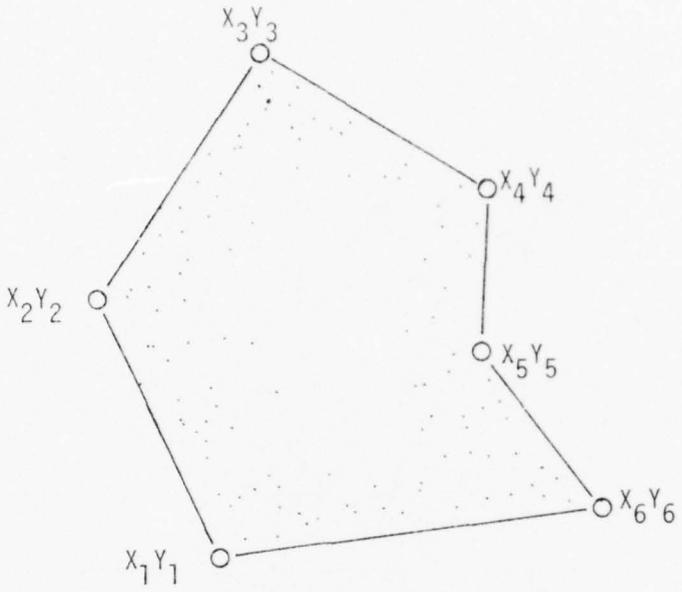


Figure 26. Boundary description of a shape.

Boundary models have been popular in shape description systems. The BUILD[47] and EUKLID[48] systems for mechanical part description, the GEOMED[49] system for scene analysis, and the BDS[50] system for building description use boundary models.

[47] I. C. Braid, "The Synthesisis of Solids Bounded by Many Faces," Communications of the ACM, Vol 18, No. 4 (April 1975).

[48] M. Engeli, EUKLID - Eine Einfuehrung, (Fides Rechenzentrum, Zuerich, 1974).

[49] B. G. Baumgart, Winged Edge Polyhedron Representation, Stanford Artificial Intelligence Project Memo AIM-179 (Stanford University, October 1972).

[50] C. Eastman, "General Purpose Building Description Systems," Computer Aided Design, Vol 8, No. 1 (January 1976).

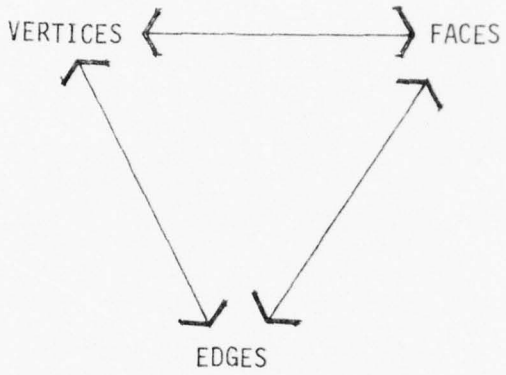


Figure 27. Conversions between vertices, faces, and edges.

3.1.5 Geometric Discipline and Shape Accuracy

The various classes of objects that one might wish to model using a geometric description system may be classified (Figure 28) according to whether they are

a. Two-dimensional or three-dimensional

b. Polyhedral or continuously curved

c. Composed of many or few parts.

Geometric modeling software for dealing with artifacts in different classifications differs in character, since different technical issues become critical. Mapping systems are dominated by the need to handle large numbers of planar polygons, automobile

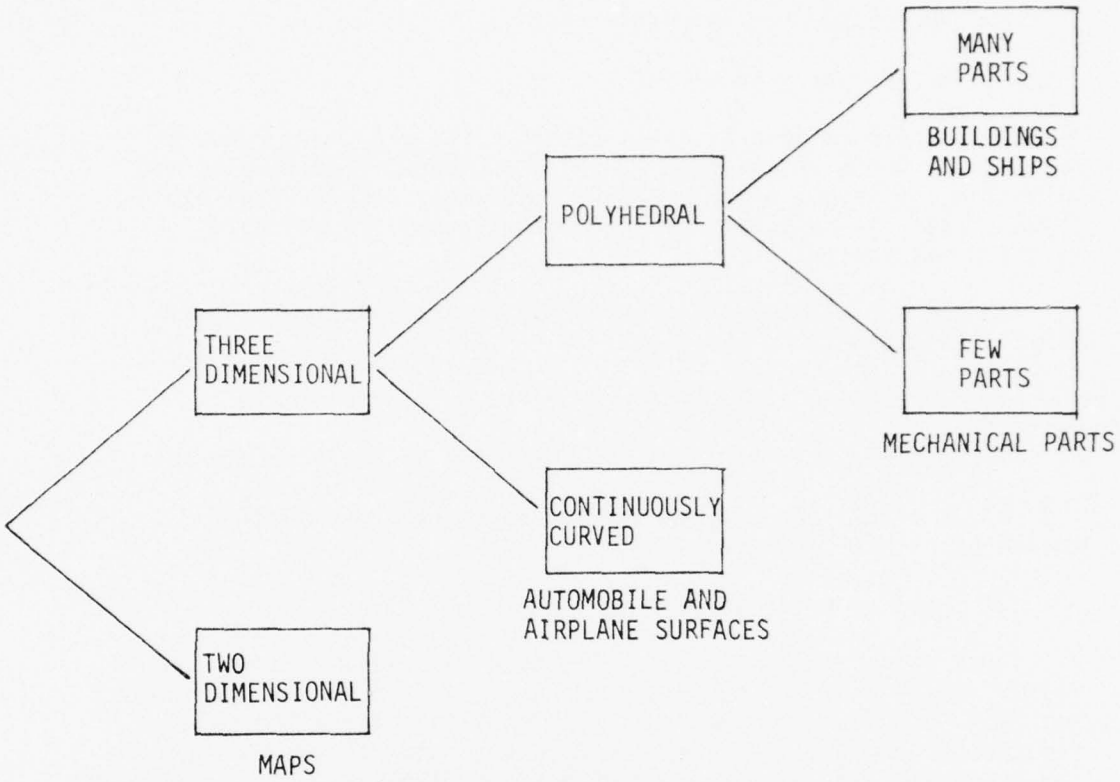


Figure 28. Classification of artifact geometry.

and airplane surface description systems by curve description issues, mechanical part description systems by the special problems of complex many-faced polyhedral objects, and building and ship description systems by the necessity to deal with large numbers of mostly fairly simple polyhedra in a complex hierarchical structure. It is unreasonable to expect software developed for dealing with one category of artifact to be very effective for dealing with another category. In particular, mapping and curved-surface description systems are not likely to be useful for building description. Ship and mechanical part description systems, since they are quite closely related, may possibly be adaptable for building description.

Closer examination of the polyhedral parts of a building permits the following distinctions between different types of polyhedra:

a. Planar or curved faces

b. Convex or nonconvex

c. Rectilinear or nonrectilinear

d. Without or with holes.

Taking the combinations of these distinctions yields 16 types of polyhedra. The simplest type is planar, convex, rectilinear, and without holes (e.g., a two-by-four). The most complex type has curved faces, is nonconvex and nonrectilinear, and has holes (e.g., a wash-basin).

In general, more complex types can be approximated by less complex (Figure 29):

a. Curved faces can be approximated by planar facets.

b. Nonconvex forms can be approximated by their convex hull.

c. Nonrectilinear forms can be represented by bounding rectangular parallelepipeds.

d. Holes can be ignored.

All these approximation methods have been very widely used. Use of planar facets to approximate curved surfaces is a standard computer graphics technique. Convex hull approximations are often used in hidden line removal, interference checking, and pattern recognition algorithms.[51] The bounding rectangular parallelepiped approximation has often been used in building description, e.g., by IMAGE[52] and OXSYS.[53]

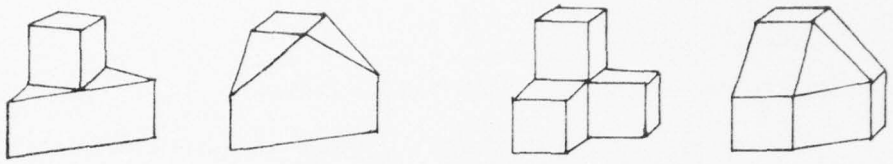
For many practical architectural purposes, use of an approximate shape description is entirely adequate, and the expense of creating and manipulating a more accurate description is not justifiable. However, use of shape approximation where appropriate should be at the discretion of the user; it should not be imposed by the geometric description software. This implies that the general geometric description software of the CAEADS data base system ideally should be able to handle

a. Compositions of many polyhedra

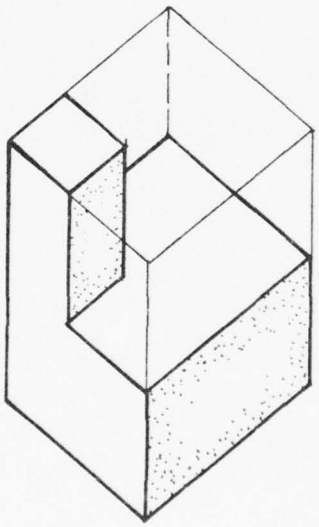
[51] A. Appel and P. M. Will, "Determining the Three-Dimensional Convex Hull of a Polyhedron," IBM Journal of Research and Development (November 1976).

[52] G. Weinzapfel and S. Handel, "IMAGE: Computer Assistant for Architectural Design," in C. Eastman (ed.) Spatial Synthesis in Computer Aided Building Design (Wiley, 1975).

[53] P. Richens, "Geometry and Numbers in Building Systems," in D. Hawkes (ed.), Models and Systems in Architecture and Building (The Construction Press, 1975).



(a) OBJECTS AND CORRESPONDING CONVEX HULLS



(b) BOUNDING RECTANGULAR PARALLELEPIPED

Figure 29. Approximating the shapes of polyhedra.

b. Some common types of curved, as well as planar, faces

c. Nonconvex and nonrectilinear forms

d. Holes.

3.1.6 Dimensional Accuracy

There are two aspects of dimensional accuracy:

a. Whether the designer currently knows precisely what a dimension should be

b. Whether the software system can represent a dimension to the required level of accuracy.

The first aspect arises because building design normally proceeds from rough sketch concepts to precise final concepts. At the sketch stage, requiring that every dimension be specified with precision is inappropriate. Instead, dimensions are better expressed as a range over which variations may take place. Thus, it is desirable for a building description system to allow for both exact specification of dimensions and expression of dimensional constraints by means of inequalities.

The second aspect arises because a word in computer memory can only represent a number with limited precision, and more critically, because input devices like digitizers allow only limited precision. Achieving sufficient precision of internal representation does not raise any particular difficulties. The limited precision of digitized input, however, means that it is unsuitable for direct use for many purposes. Either it must be processed in some way before use (see section 3.3), or other input modes must be relied upon for precise expression of dimensions.

3.1.7 Nongeometric Properties

The concept of attaching nongeometric properties to geometric entities is straightforward, but in practice there are complications. First, predicting the number and type of nongeometric properties that a designer might wish to assign to a design element is not generally possible. Unless some very strong assumptions are made about the types of applications to be carried out, it is not satisfactory to employ a fixed template (such as is provided by the DMLs of many data base management systems) in which non-

geometric properties are "filled in." Instead, some fairly sophisticated scheme of variable-length, self-describing records must be used.

Second, the dynamic character of the building descriptions raises questions about what happens to nongeometric properties when geometry is altered; for example

a. If an extra face is added to a polyhedron, should that face have the same nongeometric properties as the other faces?

b. If polyhedra are located with faces touching, what happens to the nongeometric properties of the resultant joint face?

Some systematic scheme is necessary for updating nongeometric properties as geometry is manipulated. It does not appear that any architectural computer aided design system developed so far has made a serious attempt to solve this problem.

3.1.8 Expression of Relations

The way in which the various spatial and systematic relations which exist between entities in a building description are expressed largely depends upon the view of the world which the implementation software imposes. A CODASYL-style data base management system expresses a network of relations by a network of pointers. Childs'[54] set theoretic data structure expresses relations by means of set operations. Codd's[55] relational data bases express relations in tabular form. A procedural model coded in FORTRAN expresses relations algebraically.

The task of building description requires powerful facilities for expression of relations. The following general approach is recommended:

a. Use several hierarchical indexing schemes to express gross spatial and functional groupings of elements (see section 3.2 for details)

b. Provide a general facility for creation of link records to structure the data in additional ways as required

c. Provide a general facility to allow any collection of objects

[54] D. L. Childs, Description of a Set-Theoretic Data Structure, CONCOMP Technical Report No. 3 (University of Michigan, 1968).

[55] E. F. Codd, "A Relational Model of Data for Large Shared Data Banks," Communications of the ACM, Vol 13, No. 6 (June 1970).

to be associated together and named as a set

d. Express the complex details of interrelation and interaction procedurally.

The overall hierarchical indexing provides access efficiency. Structures of link records are a useful way of expressing certain types of relations, such as adjacencies between spaces. The ability to create and refer to sets is an essential aid to the designer. The procedural modeling approach is flexible and general enough to express very complex relations. These concepts are discussed in detail in the following section.

3.2 FILE AND PROGRAM STRUCTURES

3.2.1 Indexing the Component Catalogue File

The natural way of indexing a project component catalogue file is to classify components into families and subfamilies in the way that they are normally thought of by architects and engineers. This reflects broad functional groupings and the broad partitioning of tasks among different types of application programs. A typical example of this approach is the OXSYS component CODEX,[56] which indexes components as illustrated in Figure 30. Experience with OXSYS suggests that a three-level hierarchy is sufficient for the relatively small catalogue files needed when working with a component building system. For larger catalogues, as would be needed in CAEADS, one or two extra levels would probably be necessary.

3.2.2 Indexing the Building Project File

Several types of indices are likely to be required for the building project file. The combined experience of OXSYS, CEDAR, interior space planning systems, and ship design systems suggests that at least the following should be considered:

a. Spatial indexing

b. Zonal indexing

c. Indexing by administrative category

d. Cross-referencing to and from component catalogue.

56 P. N.Richens, OXSYS-BDS User's Manual (Applied Research of Cambridge Ltd., February 1977).

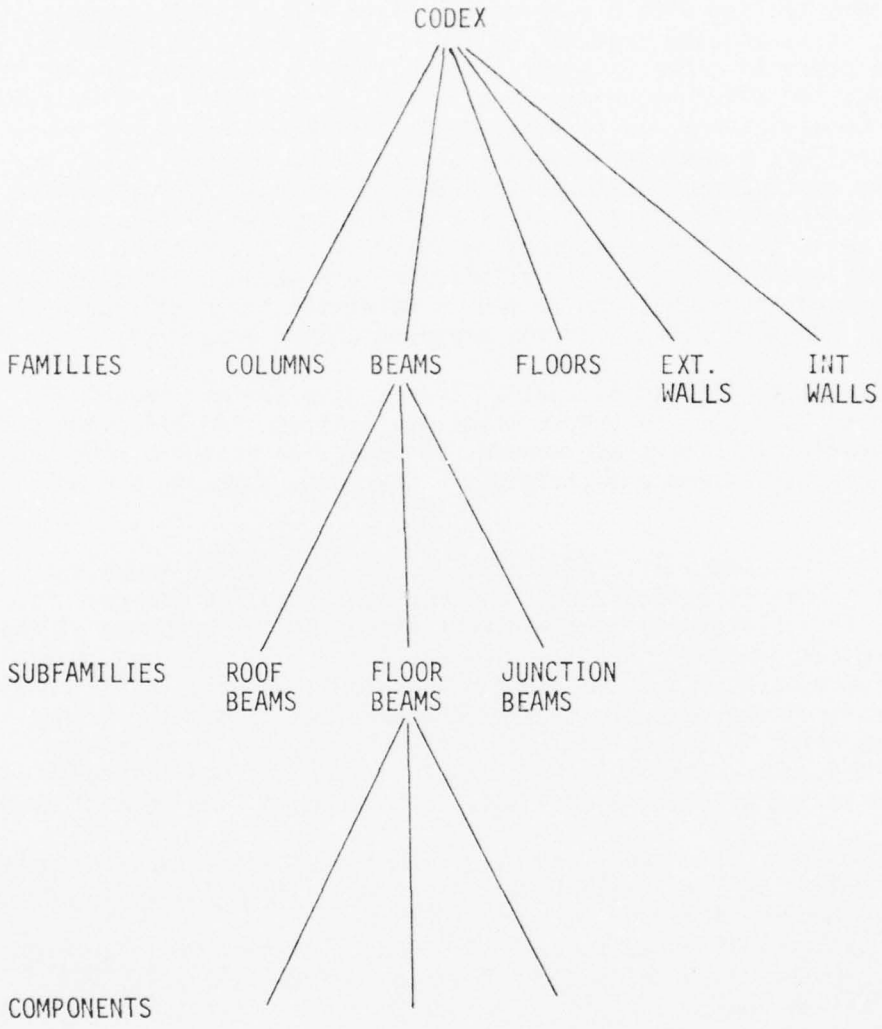


Figure 30. OXSYS CODEX indexing.

Spatial indexing is a certain requirement, so that records describing items can be accessed by item location. The simplest approach to spatial indexing (as used, for example, in OXSYS) is to subdivide the volume surrounding the building into cubic cells and to index the items falling within each cell. If an item intersects several cells, it is indexed from all of them. Big cubes can be divided into little cubes in order to generate a multilevel indexing system, if desired. An alternative approach, which seems likely to have advantages in efficiency, is to dynamically partition the global space and construct a tree-structured index as items are spatially located. This is the approach followed in CEDAR.[57] Various alternative methods for handling the special case of the item which intersects several cells have been described by Chalmers,[58] Eastman and Lividini,[59] and Charlesworth.[60] Careful calculation, and perhaps experimentation with alternatives, will be needed to determine the best type of spatial indexing for use in the proposed CAEADS data base.

Even with the aid of spatial indexing, a certain amount of searching is needed to access items by location, and efficient spatial search algorithms are needed. Discussions of these have been published by Eastman and Lividini,[61] Chalmers, Sampson, and Webster,[62] and Baxter.[63]

The efficiency of access provided by the spatial indexing/search scheme is a highly critical consideration for two reasons. First, it is likely to have a direct effect on the response of the interactive graphic interface (see section 3.4.5). Second, manipulation of a building description repeatedly requires solution of some form of the general interference problem. The interference problem, which is the determination of how polygons or polyhedra intersect or overlap, manifests itself in various special forms as the windowing problem in graphics, as the hidden line removal prob-

[57] J. Chalmers, P. Sampson, and G. J. Webster, "Data Structures Used in CEDAR," in M.A. Sabin (ed.), Programming Techniques in Computer Aided Design (NCC Publications, 1974).
[58] J. Chalmers, "The Development of CEDAR," International Conference on Computers in Architecture (British Computer Society, 1972).
[59] C. Eastman and J. Lividini, Spatial Search, Institute of Physical Planning Research Report No. 55 (Carnegie-Mellon University, May 1975).
[60] D. J. Charlesworth, Spatial Organization: A Set-Based Method of Representing Relationships Between Spaces (Property Services Agency, London, 1976).
[61] Eastman and Lividini, Spatial Search.
[62] Chalmers, Sampson, and Webster, Data Structures Used in CEDAR.
[63] R. Baxter, Computer and Statistical Techniques for Planners (Methuen, 1976).

lem, in executing the spatial set operations (see section 3.3.4), and in checking for spatial conflicts between components located in a building. Where any large number of elements is involved, solving the interference problem by some sort of exhaustive comparison procedure becomes particularly expensive. However, if an efficient spatial indexing/search facility is available, as many irrelevant objects as possible can be quickly and easily eliminated by culling, and the relatively few objects remaining can be subjected to more detailed and expensive testing.

Zonal indexing, a concept implemented in the OXSYS system, provides a useful facility to the designer. An architect commonly thinks of a building as subdivided into various types of zones -- functional zones, service zones, fire compartments, etc. Each zone may be divided into a hierarchy of subzones -- block, floor, and room. These zone hierarchies may be defined at the briefing stage if desired, i.e., before items are spatially located. OXSYS provides facilities for user definition of several different types of zone hierarchies as appropriate to the project at hand and for indexing of items according to their zone.

Indexing by administrative or departmental category is a similar concept to zonal indexing. Whereas a zone or subzone is assumed to be defined over a contiguous area, an administrative category is defined over a set of spaces which fall within some administrative grouping. These spaces are not necessarily contiguous. Since programs of accommodation requirements are often organized according to a hierarchy of administrative categories, indexing and accessing in this way is often convenient.

Assuming that the component catalogue is classified by functional families and subfamilies as described previously, pointers from the component catalogue to the project file provide indexing of the project file by component types (however the overhead incurred by this type of indexing may not be justified). Conversely, pointers back from the project file to the component catalogue give an indexing of the component catalogue file by location or zone.

Figure 31 illustrates the overall basic structure of indexing required for building project files and catalogue files. The structure corresponds quite closely to the structures employed in both OXSYS and CEDAR.

3.2.3 Indexing the Site File

Site data naturally break down into a number of relatively independent categories:

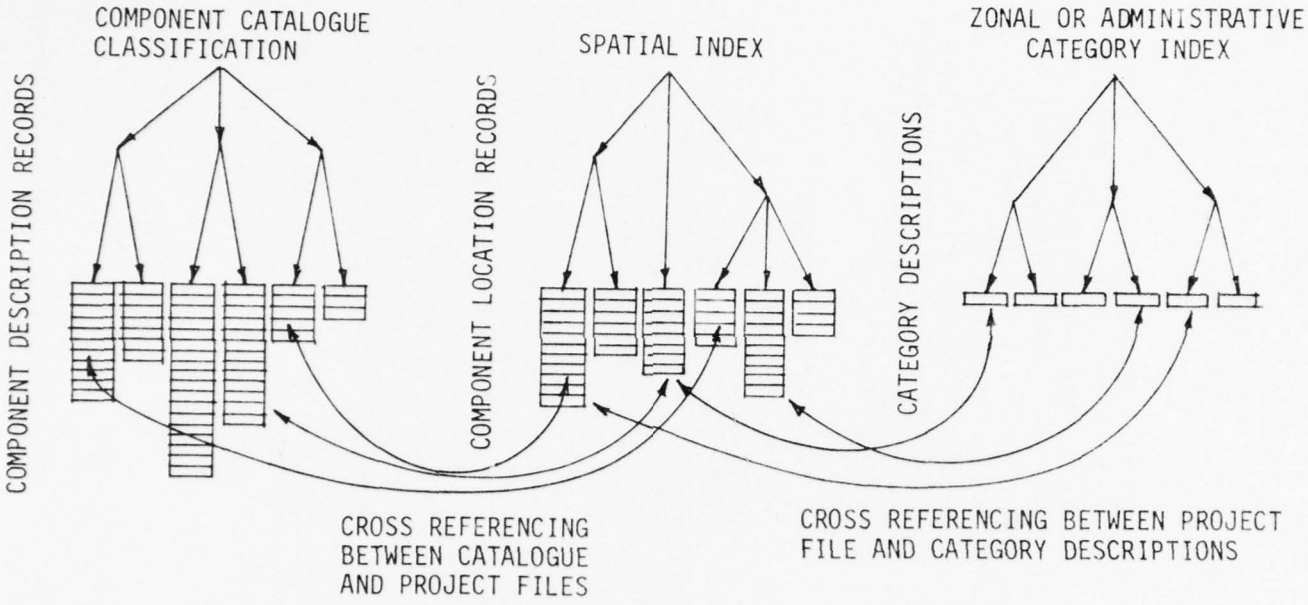


Figure 31. Indexing of files.

a. The ground model

b. Utility networks

c. Surface features (roads, paving, groundwater boundaries, etc.)

d. Vegetation

e. Existing buildings.

These categories provide a basis for overall classification of data in the site file. In addition, site data should be indexed in the same way as building description data, i.e., spatially, zonally, and by component type (where applicable).

The ground model presents some special problems, since the geometry of a topographical surface generally differs radically from the geometry of building elements. It is a complex $2\frac{1}{2}$-dimensional surface which must be described by a set of data points rather than some curve formula. The following are a number of alternative ways of holding this kind of data:

a. Random data points

b. Data points on a regular grid

c. Contours

d. A triangulated network.

No one method is suitable for all applications. The system should allow surface description data to be held and indexed in any of these forms and provide conversion routines as shown in Figure 32.

3.2.4 Sets

Most drafting systems include a facility for associating standard graphic entities into macros, which can then be named and manipulated as units. A similar facility is needed in a building description system so that subassemblies of elements (spaces and/or physical components) can be named and manipulated as units when required.

This type of set definition facility is explicitly provided in Eastman's BDS. In some other systems, sets are implicitly created when subassemblies are named and placed in the project catalogue.

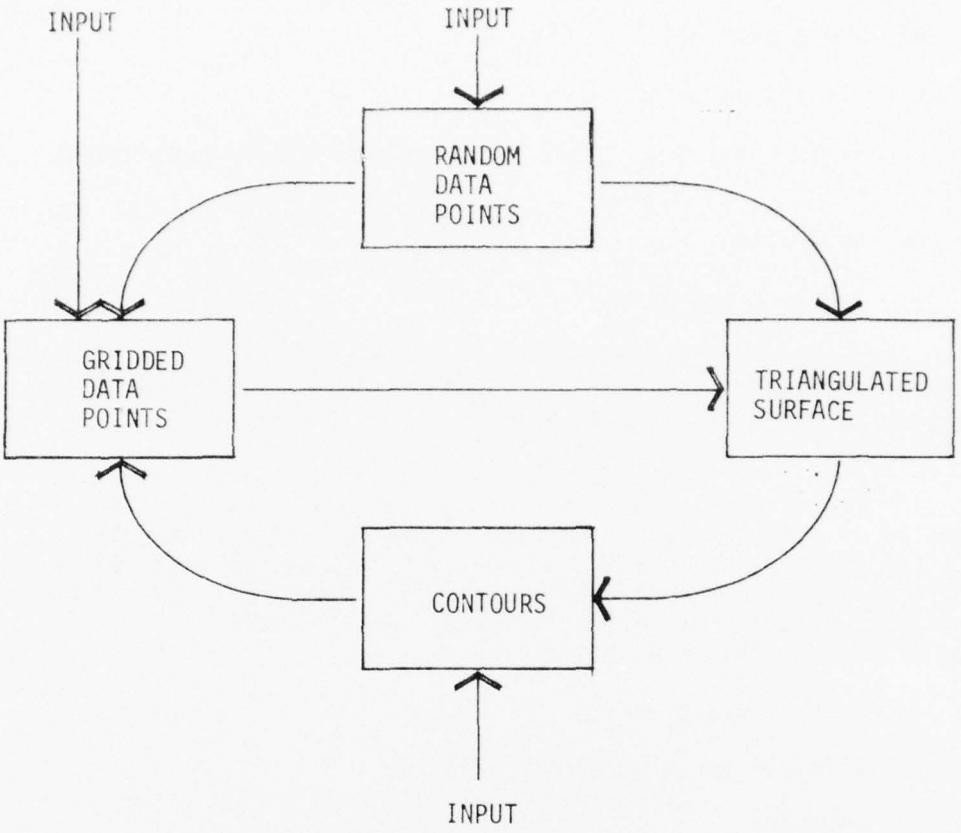


Figure 32. Conversions of the ground model.

3.2.5 Additional Link Records

In addition to the basic structure of hierarchical file indexing and sets, providing a facility for creating additional structures of link records as required is useful. A link record which is a CODASYL concept, relates two other records and describes the nature of the relation. Thus, link records might be used, for example, to encode the adjacency graph of a floor plan by connecting records describing adjacent rooms. Maintaining link records for this type of purpose may sometimes prove worthwhile.

3.2.6 Record Structures

In describing a building, the types of attributes which a user may wish to assign to a design element, the sequence in which attributes will be assigned, and the quantity of information needed to describe an attribute are not generally predictable. Therefore,

the simple fixed template record concept commonly used with CODASYL-style data base management systems is not generally adequate. Any attempt to impose some discipline of attribute assignment will make the system unwieldly and annoying to use.

In response to this requirement, advanced architectural CAD systems employ sophisticated record structures. For example, OXSYS is implemented using the plex-like record structures provided by OXSYS/BOS.[64] Eastman's BDS[65] employs a record structure in which attributes have a name, a value, and a type (like FORTRAN variable). The four attribute types used are

a. NUMBER (real)

b. CHARACTER (string)

c. SET (consisting of pointers to all elements having a given attribute value)

d. FUNCTION (takes as value an expression which can be evaluated each time it is called).

3.2.7 The Structure Versus Search Question

There is an overhead attached to the types of access facilities that have been described. The costs of necessary storage and updating of indexes may become prohibitive. The alternative is to organize records in some simple structure and expect to perform more searching through the structure in order to access a record. There are many alternatives between the two extremes, and an optimum trade-off with respect to a particular pattern of data usage must be sought.[66] Because it is difficult to determine this trade-off in advance for complex computer-aided design applications, the best approach is to provide facilities for creating as much or as little structure as proves necessary.

---

[64] OXSYS-BOS: Short Technical Description (Applied Research of Cambridge Ltd., May 1976).

[65] C. Eastman, "General Purpose Building Description Systems," Computer Aided Design, Vol 8, No. 1 (January 1976).

[66] A. Baer, C. Eastman, and M. Henrion, A Survey of Geometric Modeling, Institute of Physical Planning Research Report No. 66 (Carnegie-Mellon University, March 1977).

3.2.8 Physical Organization of Records in Storage

Depending upon the type of implementation software that is chosen, data will be passed between core and disk in one or another of the following ways:[67]

a. The file is explicitly divided into a number of segments, and the swapping is explicitly under control of application programs.

b. A virtual memory is employed, divided into a number of pages. Space allocation and swapping of pages are transparent to the application programmer.

c. Some combination of the above.

In any case, efficiency demands that exchanging of data be minimized. This can be achieved by physically grouping together records that are likely to be needed in core together. Experience with OXSYS and CEDAR suggests that the following general strategy is appropriate for building description data bases:

a. Physically group component catalogue records according to the functional type indexing scheme described previously

b. Physically group records in the building project file by spatial index

c. Physically group site records first by data category, as outlined previously, then by spatial index.

It should be pointed out that there is room for legitimate disagreement as to whether this essentially spatial organization of data is most appropriate, or whether a scheme more oriented toward optimizing access by nonspatial routes would be better. The conclusion stated here is based upon the considerable experience of several development groups that have implemented architectural CAD systems, and upon the assumption that response of the interactive graphic interface and the ability to efficiently solve the interference problem in its various manifestations are likely to prove highly critical. However, the only way to settle the question for CAEADS is by careful and detailed simulation of the system in operation.

67 R. J. Hubbold, "Multi-Level Data Structures, Segmentation and Paging," in M. A. Sabin (ed.), Programming Techniques in Computer Aided Design (NCC Publications, 1974).

3.2.9 Program Structures

Just as records holding data in a file are embedded in some kind of structure, so too are the statements and algorithms which procedurally encode relations between entities in a building description. In this case, the elementary units of the structure are individual statements, procedures, or subroutines.

Where a strongly procedural approach to geometric modeling is taken, as advocated here, the problem of data structuring is matched by an analogous problem of program structuring. This has several aspects:

a. Ensuring that the structure is coherently organized and comprehensible. Attendant upon this are an appropriate language syntax and adherence to the principles of structured programming.

b. Providing efficient access to code as needed for execution. An overlay system is one way of achieving this end.

c. Allowing convenient and rapid creation and modification of program structures modeling particular systems. An APL-like interpreted language is one way of providing this type of facility.

Unfortunately, FORTRAN used with a conventional operating system is far from an ideal medium for building the kinds of program structures needed for geometric modeling. OXSYS/BOS substantially mitigates these disadvantages by embedding FORTRAN in an operating system which facilitates implementation of overlay structures and allows independent compilation and then relinking of individual overlays. Powerful, interpreted ALGOL-like languages such as EULER and GLIDE represent another promising approach. The Evans and Sutherland Design System Language [68] takes a third very interesting direction.

Since the principles of the Evans and Sutherland language are rather unusual, a brief description of the language is given here. Basically, the system has an operand stack and a large and extensible library of named operators. An input line can consist of data, the name of an operator, or an expression consisting of both. When the system encounters an item of data, it puts it on top of the operand stack. When it encounters an operator, it executes the corresponding procedure on the data at the top of operand stack and pushes any resulting data onto the top of the stack. Extension of the library of operators is easily accomplished by creating a sequence of operators and naming the sequence as a new operator. The names of operators are stored in dictionaries, and

[68] The E & S Design System (A Brief Preliminary Description)(Evans and Sutherland Computer Corporation, August 10, 1976).

the construction of a dictionary as implemented here provides a powerful means for organizing and accessing operators. This brief description does not do justice to the apparent power and flexibility of the language, but it does give some idea of its general character.

3.2.10 Binding Strategy

Whenever part of a description is defined procedurally, the question arises as to when the values of derived variables should be computed (bound). This is a question of tuning the system for optimum efficiency in response to a particular pattern of access. The basic options are

a. Perform procedural expansion when parameters are entered and store data in expanded form

b. Perform the expansion when an instance of an element is located in the project description

c. Perform the expansion whenever an element is moved into core

d. Perform the expansion immediately prior to a computation for which the expanded data are required.

Control of binding can be

a. According to a fixed strategy built into the system

b. Assigned to user programs

c. By user command.

The best way to handle binding in a three-dimensional building description still appears to be an open question, and there is little published discussion of the topic. One piece of evidence is provided by Eastman's BDS, however, in which the following strategy was found to be optimal: "Predefined elements are stored in a parts catalogue in terms of topologies, subroutines, and values. The values are stored in tabular form. When selected for a project file, the routine and values are combined to derive the constant shape dimensions and attributes collected in the form file. New attributes may be added, if desired, at either the form or location levels. In core, two alternative representations are available, a compacted topology which accesses vertex coordinates or an expanded shape definition." [69]

69 C. Eastman, "General Purpose Building Description Systems," Computer Aided Design, Vol 8, No. 1 (January 1976).

3.3 INPUT OF GEOMETRIC DATA

3.3.1 Geometric Data Input Modes

To create any of the kinds of internal geometric models that have been described, geometric information must be entered into computer memory in some way. The basic input modes that can be employed are

a. Batch entry

b. Command language

c. Menu

d. Programming language

e. Two-dimensional position input

f. Three-dimensional position input

g. Optical scanning.

Batch entry of geometric data usually involves hand coding of geometric data from drawings onto coding forms, keypunching the data from the forms, and batched input of the punched data. In the past, batch entry of geometric data was sometimes used in such applications as structural frame analysis, automated drafting of working drawings, and perspective production. However, it is an inherently cumbersome, inefficient, and error-prone technique. Since inexpensive computer graphic systems have become available, the batch entry approach must be considered completely outmoded.

Employing a command language, the user types in simple commands plus parameters at a keyboard. This very common approach is quite adequate for many purposes. Descriptions of numerous computer-aided design command languages are given in a recent set of conference proceedings.[70] In architectural computer-aided design systems, the two most highly developed command languages appear to be those of OXSYS and CEDAR.[71]

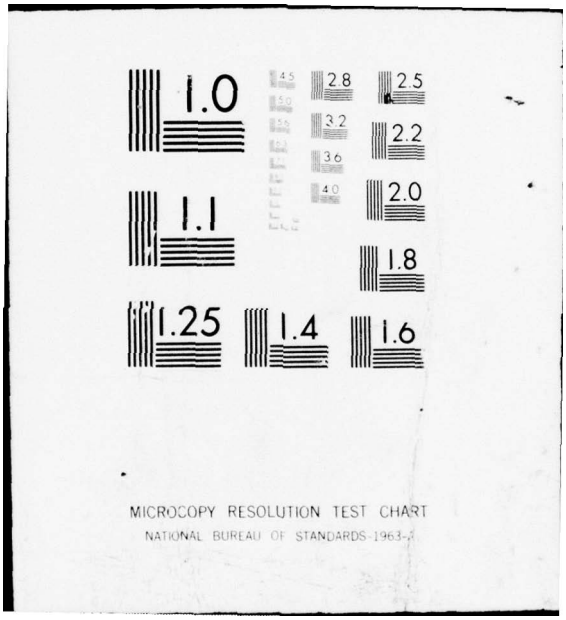
There is little logical difference between a command language and a menu, which consists of a list of commands displayed on a screen or taped to a tablet. These commands are then pointed to as required, rather than typed in at the keyboard. OXSYS has an unusually convenient and sophisticated menu system.

[70] M. A. Sabin (ed.), Programming Techniques in Computer Aided Design (NCC Publications, 1974).

[71] CEDAR 3 (Property Services Agency, London, 1977).

D-A052 040 APPLIED RESEARCH OF CAMBRIDGE LTD (CANADA) F/G 13/13
COMPUTER REPRESENTATION OF THREE-DIMENSIONAL STRUCTURES FOR CAE--ETC(U)
FEB 78 W J MITCHELL, M OLIVERSON DACA88-77-C-0001
NCLASSIFIED CERL-TR-P-86 NL

2 OF 3
AD
A052 040


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

There is no clear dividing line between a sophisticated command language and an interpreted programming language. In fact, the well-known example of APL demonstrates that a single language can sometimes be used in both modes. However, sophisticated syntax, arithmetic, iteration, control structures, procedures, etc., entitle a language to be called a programming language rather than a command language. The concept of procedural modeling of geometry as described in Chapter 2 leads to the idea that a programming language may be useful as a means for entry of geometric data in procedural form (it does not necessarily follow that the geometric data are held internally in the same procedural form).

It is unlikely that programming language input would be used very much, if at all, by ordinary project designers; however, it is an important aid in development of libraries of procedural models for use by project designers.

Two-dimensional position input is the entry of coordinates of points or more complex objects by means of some kind of automated digitizing device, most commonly a large electronic tablet. This is usually the primary means of entry of data that already exist in drawings; a good tablet or similar interface to a building description data base is essential. OXSYS incorporates a good example of such an interface.

Over the years, a number of devices for direct entry of three-dimensional coordinates have been developed, e.g., the Lincoln wand, the three-wire wand, and the Twinklebox.[72] However, although the idea of direct three-dimensional input is attractive, none of the available devices is a practical proposition for use in a working architectural CAD system.

Optical scanning devices electronically scan drawings or photographs to enter geometric data. The technology of optical scanners has now developed to the point where reliable and efficient devices for optical input of line drawings are available. For example, Laser-Scan Ltd. of Cambridge, England, markets such a device. Most of the input is handled automatically by this device, while an operator monitors progress and intervenes whenever the system fails to handle an ambiguous situation correctly. However, these devices are very expensive, and have not yet been employed for architectural applications.

72 M. E. Newell, "Man Machine Communication in Three Dimensions," in R. E. Barnhill and R. F. Riesenfeld (eds.), Computer Aided Geometric Design (Academic Press, 1974).

3.3.2 Geometric Data Input Operations

Within these various input modes, a number of different types of geometric data input operations can be provided. The types of operations implemented in various current geometric description systems include the following:

a. Explicit definition of topology

b. Entry of points and lines

c. Location of instances of standard objects.

These may be termed low level operations. In addition, the following high level operations may be provided:

a. Sweep

b. Project

c. Parameterized shapes

d. Spatial set operations.

Finally, by specifically exploiting architectural knowledge, it may be possible to provide the following very high level operations:

a. Automated component selection and sizing by context

b. Automated component location by context

c. Automated building assembly according to the rules of a specific system.

Even more types of geometric data entry operations could doubtlessly be invented, but a description of these ten possibilities should suffice for the present.

3.3.3 Low Level Operations

The potential need for explicit definition of topology arises because a boundary description of an object may store topological and coordinate data separately (see section 3.1.4). Thus, it may be convenient to describe a topology and then treat several different objects as instances of that topology. The topological structure might be entered in the form of an adjacency or incidence

matrix, or some kind of command language might be used to create edges and vertices, or a network might be drawn using the graphics interface (in this case, of course, vertex coordinates are not retained). To reduce the tedium of topology entry, some systems provide a few higher level operations. For example, Braid's BUILD[73] provides such operations as construction of a "pyramid" or a "face."

Entry of points and lines can be carried out using a command language, a menu, a programming language, or by digitization. Description of a complex object by explicit input of every vertex is an extremely laborious process to be avoided if possible. However, explicit entry and deletion of specific points and lines is an essential editing technique.

Where digitization is employed, solutions must be found to two technical problems: removal of the inevitable digitization errors, and the proper conversion of two-dimensional coordinates from the tablet into three-dimensional coordinates as required. To a certain extent, digitization errors can be removed simply by rounding tablet coordinates to the nearest location in a grid of specified interval. For this purpose, the OXSYS tablet interface allows the user to specify a grid interval appropriate to the appropriate task. A more sophisticated approach is to provide for automated latching of nearly coincident points (to make them truly coincident).[74] To remove slight misalignments between polygon boundaries, rigid or plastic merge procedures can be used. Probably the most sophisticated automatic digitization error-removal system employing these techniques in an architectural computer-aided design system is that of the Site Layout System developed by the University of Edinburgh Computer-Aided Architectural Design group for the Scottish Special Housing Association.[75]

The problem of conversion to three-dimensional coordinates is more serious. The transformation required is well known,[76] but establishing the correspondence between projections of a point on different drawings and resolving ambiguities due to coincident projections of points on the drawings is difficult. Several resolutions have been proposed, but none of them are entirely satisfactory. The Evans and Sutherland Computer Corporation markets a

[73] I. C. Braid, "The Synthesis of Solids Bounded by Many Faces," Communications of the ACM, Vol 18, No. 4 (April 1975).
[74] C. Herot, "Graphical Input Through Machine Recognition of Sketches," Computer Graphics, Vol 10, No. 2 (1976).
[75] A. Bijl and G. Shawcross, "Housing Site Layout System," Computer Aided Design, Vol 7, No. 1 (1975).
[76] I. J. Sutherland, "Three Dimensional Data Input by Tablet," Proceedings of the IEEE, Vol 62, No. 64 (April 1976).

tablet which allows simultaneous entry of points in different drawings. Lafue[77] has used theorem-proving techniques from artificial intelligence to resolve ambiguities. Thornton[78] provides a rotation facility so that ambiguity due to coincident projections of points can be avoided. Negroponte[79] has developed a sketch recognition system largely based upon scene analysis techniques from artificial intelligence.

The third kind of low level geometric data input operation, location of instances of standard objects, is a very familiar computer graphics technique. A library of standard objects such as circles, squares, furniture elements, doors, and windows, is defined. A set of transformations which can be applied to these objects is also defined; these typically include translation, rotation, reflection, and possibly scaling. A command language or menu system is then used to specify application of transformations to the objects in order to assemble a design.[80] Programming languages like Euler and Sail[81] provide very convenient facilities for manipulating instances of standard objects by means of statements like the following:

```
DOOR AT [X, Y] SCALE [S] ROT [R];
```

The standard object instantiation approach to input is widely used in two-dimensional architectural drafting and scheduling systems, typically those employed for interior space planning applications. Good examples of this type of system are SLS's MAN/MAC[82] and Morganelli-Heumann's Office Planning System.[83]

3.3.4 *High Level Operations*

The higher level sweep and project operations are closely related. The sweep operation can be used to create surfaces by swee-

---

[77] G. Lafue, Recognition of Three Dimensional Objects from Orthographic Views, Instutute of Physical Planning Research Report (Carnegie-Mellon University, 1976).

[78] R. Thornton, MODEL: Interactive Modeling in Three Dimensions Through Two Dimensional Windows, Unpublished MS Thesis (Cornell University, 1976).

[79] N. Negroponte, "Recent Advances in Sketch Recognition," Proceedings of the 1973 AFIPS Conference (1973).

[80] W. J. Mitchell, Computer Aided Architectural Design (Petrocelli-Charter, 1977).

[81] W. M. Newman and R. F. Sproull, Principles of Interactive Computer Graphics (McGraw-Hill, 1973).

[82] MAN/MAC System Description (SLS Environetics, October 4, 1974).

[83] Office Planning System (Morganelli-Heumann Inc., 1976).

ping a line through space in some specified way (Figure 33). This is a convenient and intuitively appealing way of defining ruled surfaces and surfaces of revolution. The project operation (Figure 34) can be used to create prismatic objects by moving a planar face through space. This is particularly useful for architectural applications, since many building components are prismatic. The latest version of Braid's BUILD system for geometric description[84] provides sweep and project operations.

The use of parameterized shapes is an extremely powerful concept. The basic idea is illustrated in Figure 35. The beam section is represented by the geometric model illustrated, plus the six parameters X_1, X_2, X_3, Y_1, Y_2 and Y_3. By assigning different combinations of values to these parameters, an infinite number of instances of the class of shapes "I-beam section" can be defined. It is important to note that, in general, a given shape may be parameterized in a wide variety of different ways, and each of these different ways represents an assignment of different degrees of freedom to the designer. Figure 36 shows two alternative ways of parameterizing the I-beam as an example. In Figure 36 (a), symmetry constraints are built in, while in Figure 36 (b), proportion constraints are added.

The way in which an object is parameterized can be used to control whether the effect of an operation performed by a designer will be local or global as illustrated in Figure 37. The same figure has control points disposed in different ways, so that movement of one of the control points has correspondingly different effects.

The concept of parameterized shapes is so powerful that many shape description systems rely on a very restricted vocabulary of system-defined parameterized shapes. For example, the initial version of Braid's BUILD system[85] used only the six shapes shown in Figure 38. Voelcker's PADL system[86] uses only two -- the rectangular parallelepiped and the cylinder. Many building description systems, such as Teague's BUILD,[87] CADS,[88] the SSHA house design

[84] I. C. Braid, A New Shape Design System, University of Cambridge Computer Aided Design Group Document No. 89 (March 1976).
[85] I. C. Braid, "The Synthesis of Solids Bounded by Many Faces," Communications of the ACM, Vol 18, No. 4 (April 1975).
[86] PADL Primer, Production Automation Project (University of Rochester, July 1976).
[87] L. C. Teague, "Network Models of Configurations of Rectangular Parallelepipeds," in G. T. Moore (ed.) Emerging Methods in Environmental Design and Planning (MIT Press, 1970).
[88] W. J. Mitchell, "Vitruvius Computatus," in D. Hawkes (ed.) Models and Systems in Architecture and Building (The Construction Press, 1975).

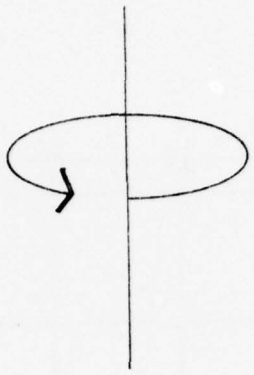


Figure 33. Sweep operation to create a cylinder.

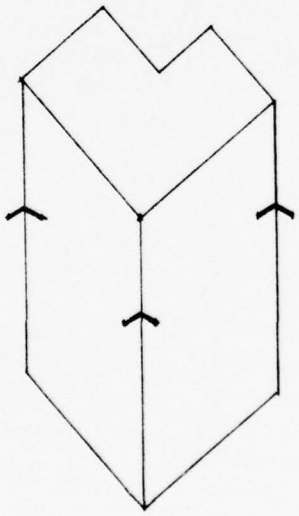
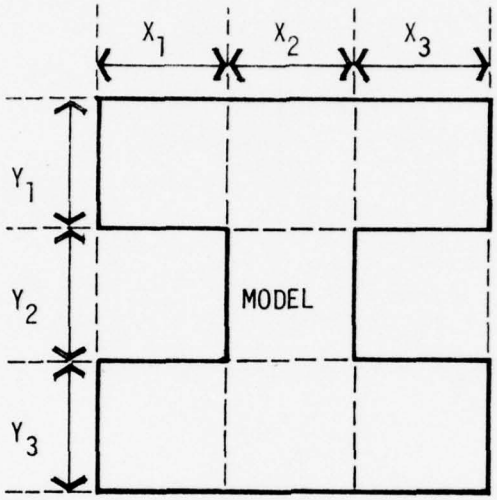
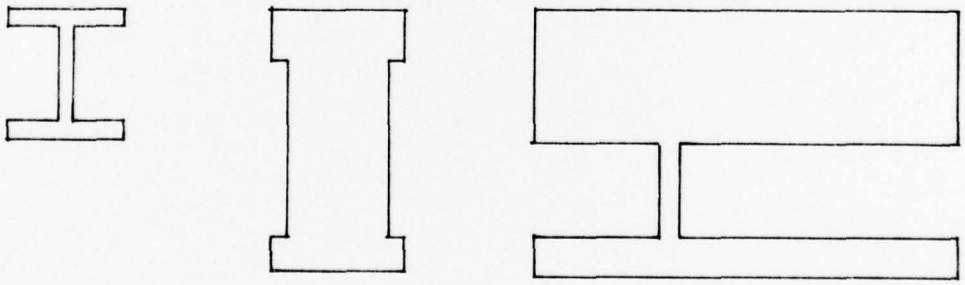


Figure 34. Prismatic object created by a project operation.

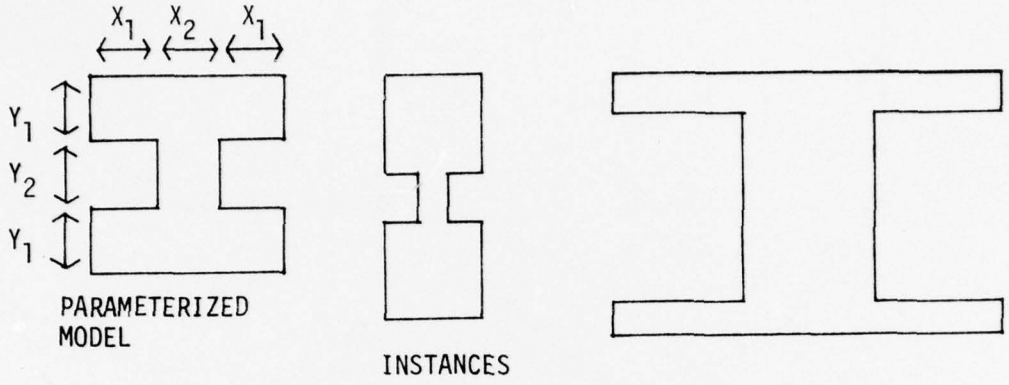


(a) PARAMETERIZED MODEL

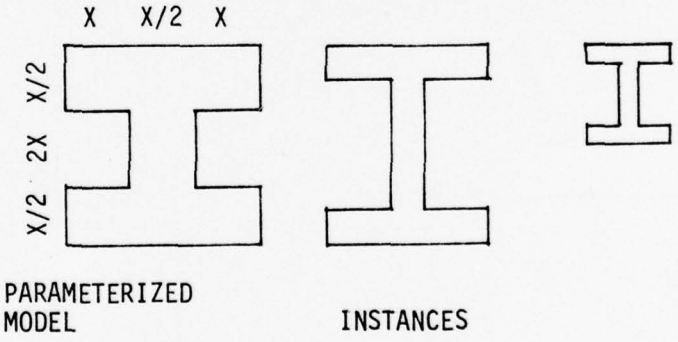


(b) EXAMPLES OF INSTANCES

Figure 35. Concept of a parameterized shape.



(a) PARAMETERIZED MODEL WITH BUILT IN SYMMETRY CONSTRAINT



(b) PROPORTION CONSTRAINT ADDED

Figure 36. Alternative parameterizations.

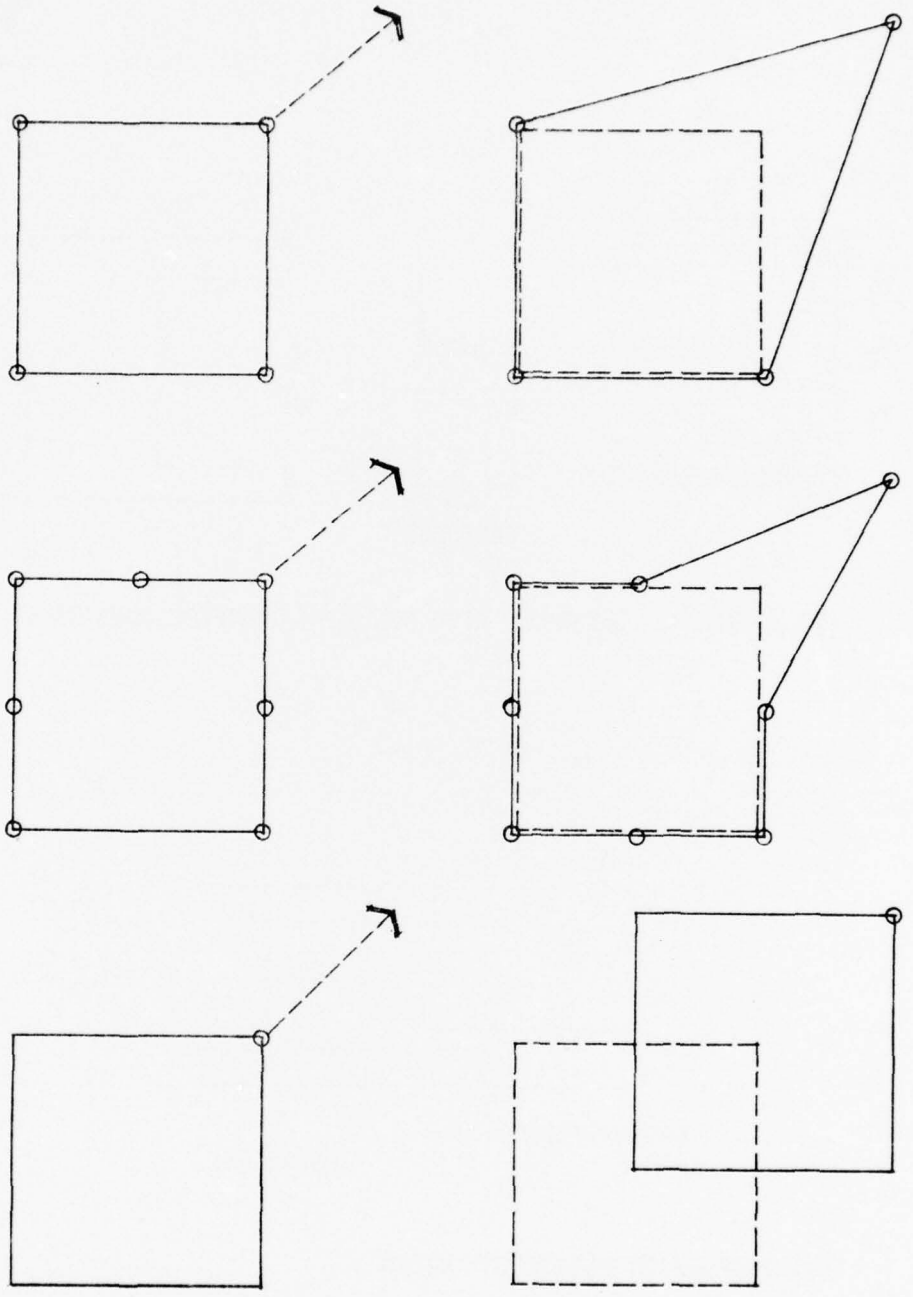


Figure 37. Effects of manipulating an object parameterized in different ways.

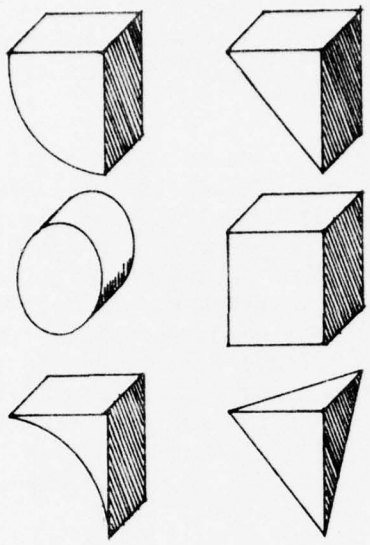


Figure 38. Primitive parameterized shapes provided by BUILD.

system,[89] CEDAR,[90] and OXSYS, have relied solely on the rectangular parallelepiped.

An exciting possibility which arises with the use of parameterized models is that it may often be possible to develop an algorithm which exhaustively enumerates a whole class of architecturally important models. For example, Mitchell, Steadman, and Liggett[91] developed an algorithm which generates all models of small rectangular floor plans. Every small rectangular floor plan possible is a dimensioned instance of one of these models. Availability of the complete set of models makes possible a systematic and rigorous approach to the optimal design of small rectangular floor plans.

[89] A. Bijl et al., ARU Research Project A25/SSHA-DOE: House Design, Edinburgh University CAAD Studies (November , 1971).
[90] CEDAR 3 (Property Services Agency, London, 1977).
[91] W. J. Mitchell, J. P. Steadman, and R. S. Liggett, "Synthesis and Optimization of Small Rectangular Floor Plans," Environment and Planning B, Vol 3, No. 1 (1976).

A programming language is needed for creation of procedures which encode parameterized shape models. At the project designer level, a simple command language or menu system can be used to create assemblies of instances describing a particular design. Using an interactive graphics interface, a parameterized object can be "sculptured" directly by using a light pen or tablet to shift control points.

The power of a shape data input system based upon the concept of parameterized models can be increased considerably by also providing the spatial set operations, i.e., some or all of union, intersection, difference, and complement (Figure 39). These can be employed to create more complex shapes from instances of the primitive models. For example, a block with a hole in it can be formed by creating an instance of a rectangular parallelepiped, creating an instance of a cylinder, locating the instance of the cylinder so it passes through the parallelepiped in the required position, and then subtracting the cylinder from the parallelepiped. Almost all shape description systems that aspire to generality provide operations for use in data input. Examples include BDS, BUILD, EUKLID, GEOMED, PADL and TIPS.[92]

For building description applications, a general complement operation may be useful as well. This concept is illustrated in Figure 40. A set of rooms is located within a building shell; their complement is then defined as the circulation space. This type of facility is implemented in OXSYS. It may also be useful for some applications to create objects composed of "empty space" by taking the complement of "solid components."[93]

3.3.5 Very High Level Operations

The types of shape description operations described so far are very general in their application and may be found in systems oriented towards such diverse tasks as mechanical part design, ship design, building design, or image synthesis. The very high level operations which will now be described are specifically applicable to architectural design description.

The first, automated component dimensioning by location, may

---

92 A. Baer, C. Eastman and M. Henrion, A Survey of Geometric Modeling, Institute of Physical Planning Research Report No. 66 (Carnegie-Mellon University, March 1977).

93 C. Eastman, "An Interrogation Language for Building Descriptions," in D. Hawkes (ed.), Models and Systems in Architecture and Buildings (The Construction Press, 1975).

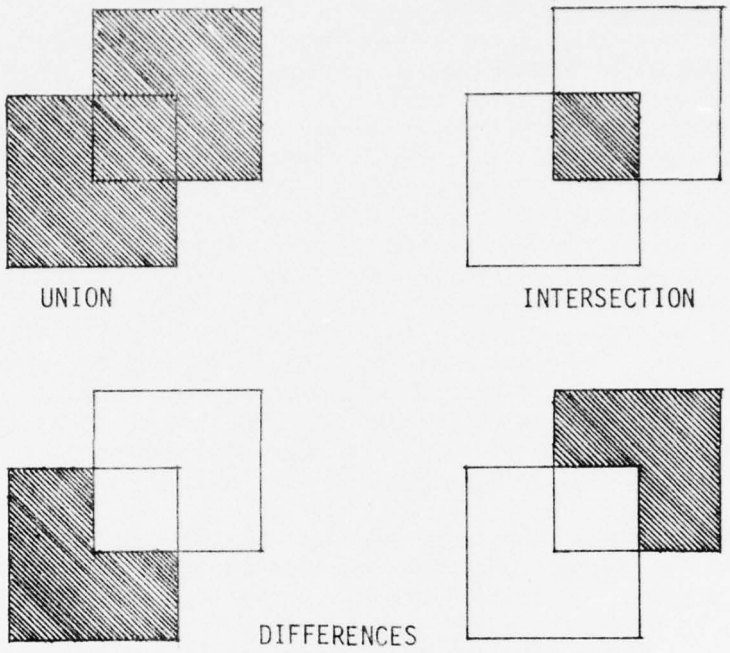


Figure 39. Spatial set operations.

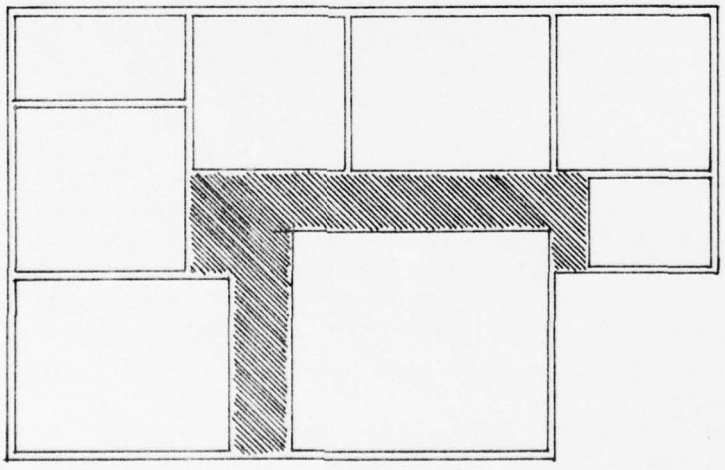


Figure 40. Use of a complement operation to create a circulation space.

be explained by reference to the previous I-beam example. As discussed, it was treated as a geometrically parameterized object. A different type of parameterization can be developed if the function of the beam is recognized to be structural. Assuming a particular material and end conditions, this function could be described by parameters for loading and span. By applying the laws of beam theory, the geometric parameters could be derived from these functional parameters. Furthermore, within a well-defined building system, the functional parameters can be derived directly from context parameters describing the beam's location within the building. Figure 41 illustrates the flow of operations in deriving a complete geometric description of the beam section. Apart from well-known, classical engineering methods, very powerful and general methods of automated component dimensioning, are provided by nonlinear and dynamic programming techniques.[94] Very extensive and powerful automated structural components dimensioning facilities have been implemented in the HARNESS[95] computer-aided architectural design system.

By a similar use of knowledge of the rules of a building system, very high level automated component location and detailing operations can be developed. The OXSYS system, for instance, automatically locates beams to support a slab, automatically fills in facade details, and details roof lights and partition corners. Moore, Brotton, and Glover[96] have demonstrated how a fairly simple automated steel frame detailing system can be implemented.

It is not difficult to write ad-hoc automated systems for handling particular types of details. More importantly though, some general mathematical theory for dealing with detailing problems is emerging. Most detailing problems can be conceived of as top-down substitution operations, and formalisms such as Stiny's[97] shape grammars can be used to describe context-determined substitution rules.

The highest level operation is automated building assembly. This operation begins with a description of the building as an assemblage of space and then employs automated component location, selection, and detailing submodels to "fill-in" the structure of components required to realize that design within a particular construction system. The same basic design might be automatically assembled according to several different construction systems, and

[94] W. J. Mitchell, Computer Aided Architectural Design (Petrocelli-Charter, 1977).

[95] J. Jacobsberg, "Computer Design Aids for Large Modular Buildings," in D. Hawkes (ed.), Models and Systems in Architecture and Building (The Construction Press, 1975).

[96] M. G. Moore, D. M. Brotton, and F. Glover, "Beam Details for Steel Framed Buildings," CAD 76 Proceedings (IPC Science and Technology Press, 1976).

[97] G. Stiny, Pictorial and Formal Aspects of Shape and Shape Grammars (Birkhaeuser-Verlag, Basel, 1975).

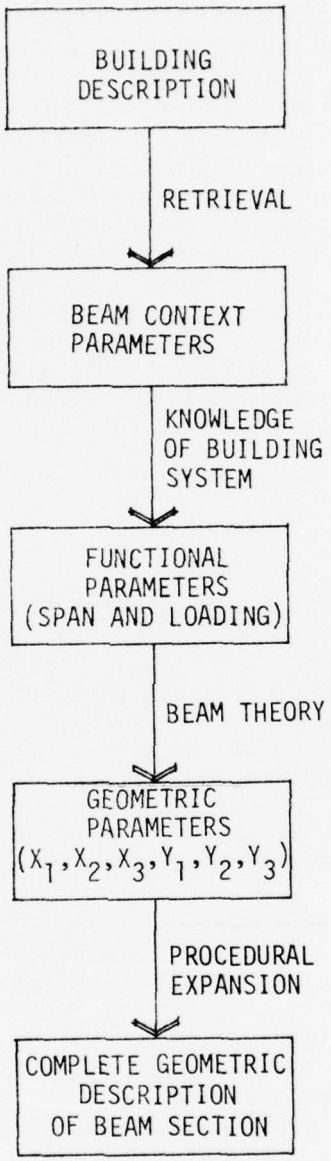


Figure 41. Example of an automated component dimensional process.

the results critically compared. This process of automated building assembly is quite directly analogous to the process of compilation of a program written in a high level language. Efficient automated building assemblers, which produce high quality results, have been implemented successfully in the HARNESS and OXSYS systems.

The very high level operations of automated selection, sizing, location, detailing, and building assembly are particularly relevant to the Corps of Engineers, since a large amount of information about design rules exists in the form of the Corps' design standards. Section 3 of Charles Eastman's report Feasibility and a Proposed Development of AEADS II[98] discusses in detail how these standards could be formalized for this purpose. It should be noted that appropriate formalization of the standards would support both very high level design definition operations and automated design checking and evaluation by CAEADS. Some investigation of methods for formalization has already been initiated within the SEARCH development effort.

3.3.6 Defaults

If a value for a design variable has not been entered, there are two choices:

a. Treat the value of this variable as undefined

b. Assume a default value until such time as this is explicitly overriden by the designer.

The concept of a default is useful, and provision should be made in a building description system for extensive use of defaults.

The role of defaults is well illustrated by the Building Optimization Program (BOP) implemented by Skidmore, Owings, and Merrill[99] some years ago. BOP derived an outline description of an office building (number of floors, floor area, number of elevators, etc.) in response to a set of parameters describing the site and the economic context. In other words, it was a simple parameterized model of an entire office building. All parameters had standard defaults, and the model could be run initially with no or very few user-defined parameters. As more information became available, additional runs could be made with more user-defined parameters.

98 C. Eastman, Feasibility and a Proposed Development of AEADS II (April 1976).

99 G. N. Harper, "BOP: An Approach to Building Optimization," Proceedings of the ACM National Conference (1968) pp 575-83.

Thus, the image of the building was brought increasingly sharply into focus as knowledge was acquired.

Among currently implemented comprehensive architectural CAD systems, CEDAR appears to make most elaborate provision for use of defaults.[100] In CEDAR, applications can be run in either "hypothetical" or "exact" mode. In hypothetical mode, relevant current default and assigned values are displayed on the screen; the user can edit these as desired, run the application a number of times with different values, and output the results to working files. When a satisfactory set of values with respect to the current application is found, the designer may attempt to assign these values within his/her working copy of the complete building model.

The essential advantage of using defaults from the user's point of view, is that a complete description of the design or a part of the design becomes available from a very early stage, so that analyses can be performed upon it. The early descriptions are not very accurate, but they become increasingly so as the concept is refined and developed in detail. From the programmer's point of view, the ability to assume that a program will encounter a default rather than an undefined value can considerably simplify application program implementation.

3.3.7 Structure of the Geometric Data Input System

Figure 42 illustrates the relations between these various types of geometric data input operations. The relationship can be viewed as a hierarchy of models, each built upon the one below. The designer/user can potentially view and operate upon a model at any level. The designer's operations are shown in the left-hand column, and the procedures needed to support these operations are shown on the right. It is essential for the success of the CAEADS system to provide the full range of types of input operations to designers. In particular, the very high level operations upon building system and building type models are vital. A system which relies entirely upon lower level operations tends to be cumbersome and expensive to use, and designers are unlikely to feel comfortable with it.

It should be pointed out that the feasibility of all these different levels of modeling has been demonstrated in implemented systems. Eastman's BDS system deals with shape primitive models, the OXSYS/DDS systems are building system models, and the HARNESS hospital design system is a very complete and detailed model of a particular type of large general hospital.

[100] CEDAR 3 (Property Services Agency, London, 1977).

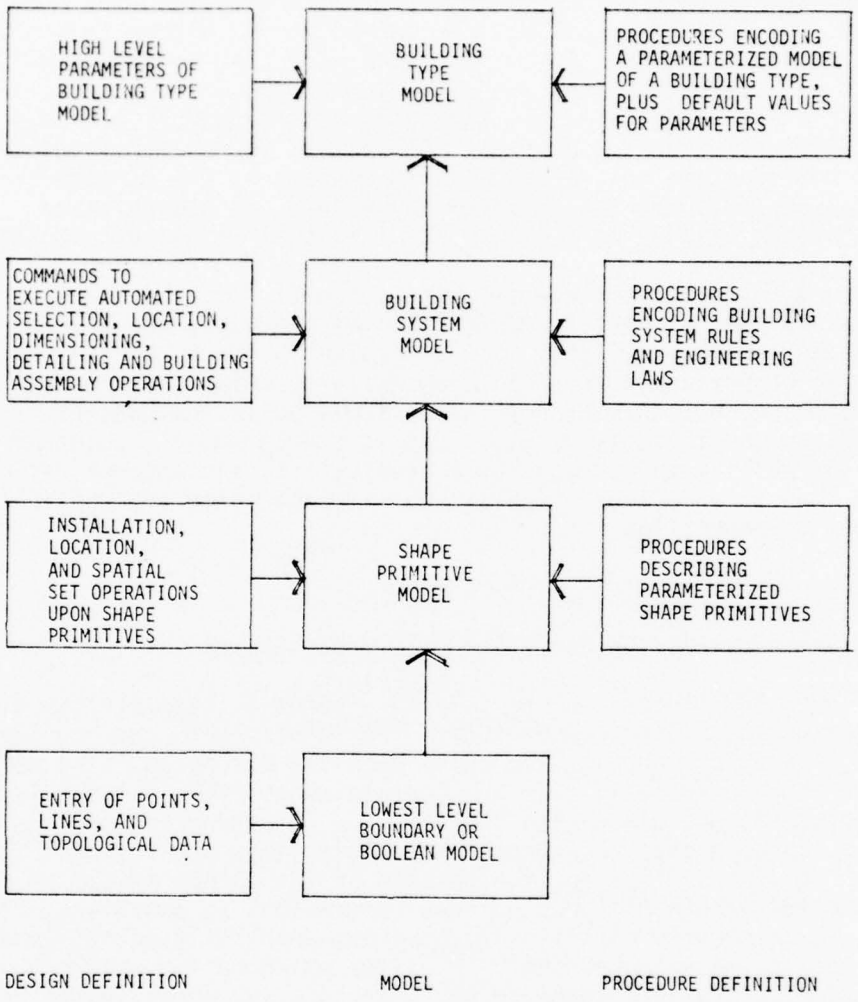


Figure 42. System of geometric data entry facilities.

Although it is possible to implement a system in which the building system and shape primitive definition procedures are fixed, this would not be satisfactory for the Corps. Provision must be made for some level of user to develop, refine, and extend these procedures as required. The building system description procedures should not be thought of as defining the type of highly rationalized, closed component building system popular in Europe. These procedures need only incorporate current accepted practice in the user office, or even of an individual designer. However, the more comprehensive, rigorous, and systematic the practices, the more powerful the procedures will be.

3.3.8 Sequencing of Input Operations

Subject to natural logical constraints, it is essential that the input system should allow maximum freedom to input data in any sequence convenient to the user. Most existing systems allow considerable freedom in a "forward" direction. However, it is much more difficult to provide a convenient general facility for deleting components or systems that have previously been located in the description and replacing them with something else. This is because an element or system, once located in a design, becomes embedded in a complex network of functional dependencies and consistency constraints.

The development of a generalized approach to building description editing, while maintaining consistency of the description, should at present be regarded as a research topic.

3.3.9 The Power Versus Generality Question

The very high level British systems that have been described are often criticized because they can only be used within rather narrowly defined contexts of building geometry, type, and method of construction. Conversely, lower level and more general systems are often criticized because they lack problem-solving power. Both these criticisms miss the essential point that a trade-off usually must be made between power and generality. Newell has stated the situation as follows: "A method has two sides. On the one hand, it demands certain information about any task to which it is to be applied. This shows up in the "givens" of the problem statement. These demands for information may be stringent or liberal. On the other hand, the method delivers certain things in the way of results... or at least chances of results... for various expenditures

of effort. Again, it may deliver a lot or a little, and it may do so with certainty or with plausibility, and cheaply or dearly. In general, the more information available the better the results that can be obtained. Conversely, strong results imply strong information demands, and weak demands can yield only weak results."[101]

The only way to achieve both power and generality is to employ the concept, repeatedly stressed in this report, of multilevel models and a multilevel system to manipulate them. The lower level facilities for input and storage of design data should be very general, and the higher level models should take advantage of information about specific problem domains to implement powerful, high level design operations. Unique buildings of unusual character can then be modeled directly using the low level facilities, while buildings of a fairly standardized type can be modeled at a very high level. Particular instances of that type can be designed very rapidly and efficiently using very high level operations. The choice as to whether a design should be created directly using low level operations or by entering parameters to an available high level model should be at the user's discretion.

3.4 OUTPUT OF GEOMETRIC DATA

3.4.1 The Process of Generating Output

The generation of output from the internal building description requires the following steps:

a. Accessing the building model to extract the required data. This may involve retrieval, procedural expansion, or both.

b. Performing any necessary transformation upon the data, e.g., generating some specified kind of two-dimensional projection.

c. Output to a graphics terminal, plotter, printer, or file.

In principle, this process allows a user considerably more freedom to specify output options than does a conventional drafting system. First, a conventional drafting system usually uses the concept of drawing "layers" to provide some measure of user control over what information is to be involved in a drawing. Using a three-dimensional data base as described here permits exploitation of the full richness of the indexing system in specifying a subset of data to be output. Second, most conventional drafting systems store data

[101] A. Newell "Artificial Intelligence and the Concept of Mind," in R. C. Schank and K. M. Colby (eds.), Computer Models of Thought and Language (W. H. Freeman, 1973).

in the form of particular two-dimensional projections and can only output exactly the same projections. A three-dimensional data base allows any type of projection with any desired parameters to be specified. Third, most drafting systems are heavily device-dependent, and many are marketed as hardware/software turnkey systems. A three-dimensional building description makes formatting data for any desired output device convenient and simple.

Detailed consideration of graphics is beyond the scope of this report, but several aspects of graphics have important implication for the data base. These are discussed briefly below.

3.4.2 Literal and Diagrammatic Graphics

When graphic output is generated by providing a projection directly from the geometric descriptions of elements stored in the data base, the result may be termed a literal picture of the building.

Alternatively, a graphic code describing a conventionalized symbol for an element might be stored with the element. When the element is to be drawn, this symbol is plotted at the appropriate location, rather than a literal projection of the element. Piping and wiring diagrams are typically produced in this way. This type of output may be termed diagrammatic graphics.

An approach to graphic output production that is between the literal and the diagrammatic has been implemented in systems like OXSYS, which approximate all building components by rectangular parallelepipeds. In this case, the projections of a component onto the parallelepiped faces are stored as shown in Figure 43. High quality plans, elevations, and sections in planes parallel to the faces can then be generated very quickly and easily, as illustrated in Figure 44. Perspectives, on the other hand, are literal projections of the parallelepiped.

Arguing whether literal or diagrammatic graphics is better is pointless. They tend to play different roles in graphic communication, and either may be appropriate for some particular purpose. Ideally, a building description system should make provision for both types.

3.4.3 Sectioning

The capability to section a building at any arbitrary plane is an extremely powerful aid to design visualization, and provision of

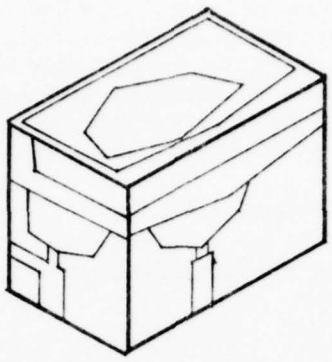


Figure 43. Projections of a washbasin onto surfaces of a surrounding parallelepiped.

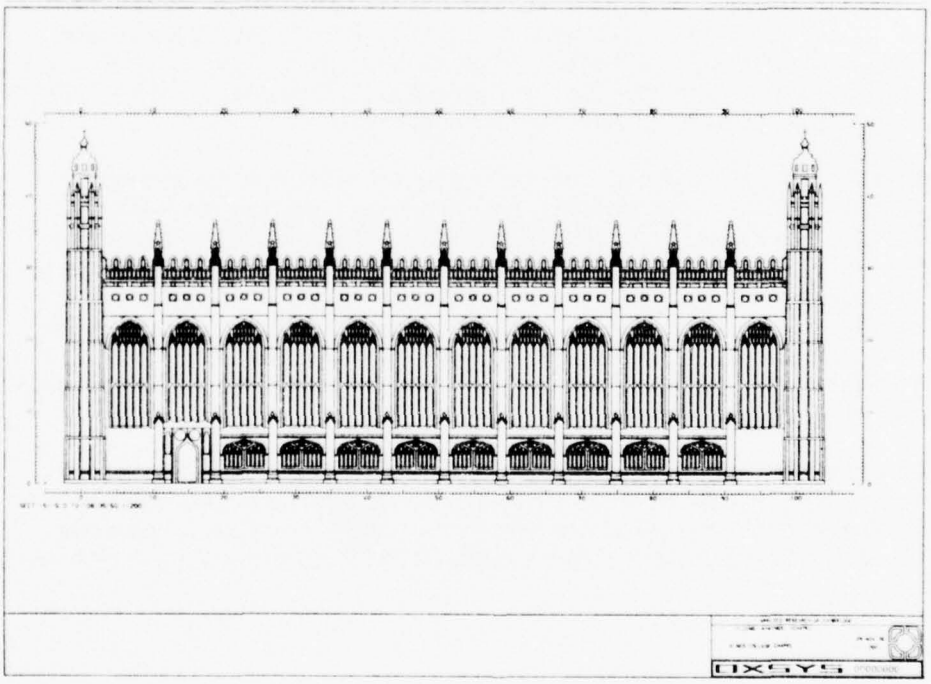


Figure 44. Elevation produced using parallelepiped method.

a sectioning facility is highly desirable. A generalized sectioning facility is only possible where a three-dimensional building description is used. Arbitrary sections cannot be produced by drafting systems which store data in the form of layered two-dimensional projections. The parallelepiped method as illustrated in Figure 43 makes generating sections to any plane parallel to the parallelepiped faces very easy, but realistic sections to angled planes are impossible.

3.4.4 Dimensioning

Since a three-dimensional building description inherently represents all the dimensions of a building, it might be thought that the problem of plotting written dimensions on a drawing would be trivial. This is not so, because the choice of reference points between which dimensions are taken requires knowledge of which particular dimensioning of an item will be useful to the architect, builder, or fabricator who will use the drawing. A draftsman has this knowledge, but a computer inherently does not. However, information about how dimensioning should be handled can be built into procedural models as a part of the building description.

3.4.5 Interactive Displays

The response of the interactive display is a vital factor in the success of the CAEADS system. Architects will use the system for design only if the display responds rapidly enough to allow fluid and uninterrupted manipulation of a design. It seems widely accepted that no more than 4/5 seconds response to such basic operations as entering a command, selecting an item from a menu, or locating and object, is tolerable. However, there is evidence that architects using a CAD system will work at a very much faster rate if the system can support this. Monitoring of tapes of an experienced user working with the SSHA system has shown a peak rate of about 30 interactions/minute, and a mean rate of about 15 interactions/minute over an extended session.

Major factors in response are the hardware configuration, operating system, and organization of the data base.

Architectural CAD systems have commonly been implemented using storage tubes and a dedicated local minicomputer. In principle, this type of arrangement can provide the level of response needed. The response delays imposed by a large time-sharing system are avoided and transmitting large quantities of data through a relatively slow line is not necessary.

Assuming that a system is physically capable of providing rapid response, the efficiency of the channel of communication between the data base and the display becomes critically important. Rapid interaction can be supported by a display file containing data base pointers or names associated with parts of the picture. When an object on the screen is indicated, the corresponding pointer is used to access the appropriate item in the data base. If (as in refreshed or raster scan displays) the display file is processed by hardware, no searching is required to find the pointer. Alternatively, if the display file is processed by software, the picture must be searched to find the pointer. However, search of the picture can be accomplished very rapidly in most cases.

An alternative approach is to use the data base's spatial indexing/search scheme as the channel of communication between display and data base. In this case, coordinates or grid references input by the user are used in a search of the data base to find the corresponding object.

3.4.6 Model Production

A complete three-dimensional description provides the potential not only to generate graphics, but also numerical control (NC) tapes. Such tapes may be used to produce models of site topography, building components, or even simplified models of complete building forms.[102] Another approach, which does not require access to specialized production facilities, is simply to write some plotter software which produces annotated "cut-outs" on sheets of paper. These are then cut and glued to produce a very accurate and effective model extremely rapidly. This type of integration of data base, graphics, and NC is becoming standard in the manufacturing industry, and it has the potential to become an important aid to architects. It is a capability that cannot be provided by conventional drafting systems.

3.5 STANDARDS AND CONVENTIONS

3.5.1 External Naming of Entities

The data indexing schemes discussed in section 3.2 have the effect of assigning several different names to each distinguished entity in a building description. The name of an entity within a particular index may be unique, or it may be shared by a number of

[102] W. J. Mitchell, Computer Aided Architectural Design (Petrocelli-Charter, 1977).

other entities. These names are used internally by the system to access entities. It is also essential to provide a facility by which a designer can access an entity of interest by specifying one of its names. The external naming system used can either be fixed and built into the software or user-defined, following some specified format.

The difficulty within the building industry of agreeing upon standard terminology suggests that the first alternative is not feasible. The system must allow for user-defined external naming.

The recommended approach to naming is well illustrated by the system provided for the OXSYS project component catalogue. At the beginning of a project, the user defines names for each family of components, e.g., columns, beams, etc. Within each family, a set of subfamily names is then defined. As each component is entered into the catalogue, it is assigned a serial number within its subfamily. Components are then identified externally by names of the following form:

< family name > : < subfamily name > : < serial number >

For example, a particular type of floor beam might be named

BEAM : FLOOR : 15

Similar approaches can be taken to zone and administrative category naming. Hierarchies can be deeper if desired, and numbers may substitute for mnemonics.

The primary means of identifying items spatially is by pointing at them on a display. However, it is also possible to name spatial cells, and to access items spatially by giving the cell name. This may be convenient for some purposes.

Advocating a user-defined approach to naming is not intended to discourage moves towards standardization of terminology. On the contrary, provision of this sort of naming discipline would provide natural encouragement for architects and engineers to standardize their terminology.

3.5.2 Metrication

Providing for potential change to metric units raises some possible problems. These can be handled satisfactorily if the principles of modular, multilevel, procedural modeling that have been advocated here are followed. Conversion routines can be in-

troduced as required at the interfaces between procedures, modules, and levels.

3.5.3 Shape Description Standards

There exists an ANSI subcommittee (Y14.26) which for some years has been concerned with "Digital Representation of Physical Object Shapes."[103] This group has developed a preliminary proposal for a standard method of communicating information about the shapes of physical components to manufacturers.

Since CAEADS is not intended for use in design of manufactured components, this proposed standard is not of direct relevance. However, it should be noted that, at some point in the future, building descriptions are likely to be delivered to contractors in digital rather than drawn form. At this stage, adoption of a standard for digital communication of building description will become an important issue.

3.5.4 Documentation

A number of different types of documentation must be systematically maintained when employing a building description data base in design. The most important of these are

a. Explanations of commands and their parameters

b. Explanation of user-defined names for component families, zone classes, etc.

c. Definitions of properties which might be assigned to entities

d. Descriptions of catalogued entities and the ways in which they are parameterized.

Since a highly flexible, user-extensible system has been advocated here, it follows that it is highly desirable to provide powerful self-documentation facilities as part of the system. This has been done in several current systems.

---

[103]American National Standards Institute, Subcommittee Y14.26, "Digital Representation of Physical Object Shapes," American National Technical Report (January 8, 1976).

The Evans and Sutherland Design system provides a particularly elegant command documentation facility. Some examples of output are illustrated in Figure 45. Figures 46 and 47 illustrate the type of documentation formats for names, properties, and components that are used in conjunction with OXSYS. The OXSYS system has the capability to store this type of documentation internally and generate documentation reports as required (Figures 48 and 49).

.DIV

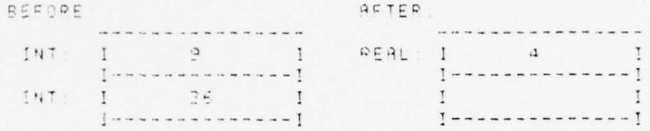
THIS COMMAND DIVIDES THE NUMBER ON THE TOP OF THE OPERAND STACK INTO THE NUMBER THAT IS NEXT ON THE STACK. THE RESULT LEFT ON THE STACK IS A REAL NUMBER.

EXAMPLE:

THE DESIGN SYSTEM INPUT LINE:

*36 9 .DIV

WILL HAVE THE FOLLOWING STACK CONTENTS RELATIVE TO THE ".DIV" COMMAND:



.DUP

THIS COMMAND DUPLICATES THE ELEMENT ON THE OPERAND STACK AND PUSHES IT ONTO THE OPERAND STACK.

EXAMPLE:

THE DESIGN SYSTEM INPUT LINE:

*(ABC) .DUP

WILL HAVE THE FOLLOWING STACK CONTENTS RELATIVE TO THE ".DUP" COMMAND:

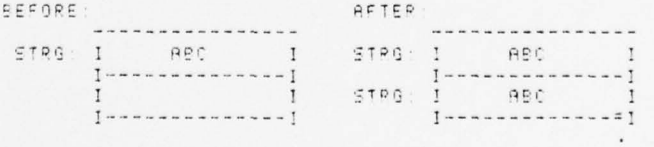


Figure 45. Examples of self-documentation of E & S Design System commands. (Reproduced with permission of Evans and Sutherland Computer Corp.)

OXSYS	PROPERTY STANDARD	
Version BDS	Usage Functional Zone Data	PM *TD
Date 19 Nov 76	Format 1R1ROX8A1R	

NAME THERMAL DATA

DESCRIPTION

Temperature and ventilation rates used in heating and cooling estimates.

VALUES

NO	F	NAME	DESCRIPTION	UNIT
1	R	DESIGN TEMP	Design temperature for heating system expressed as 'Resultant Temperature'. This combines air temperature and mean radiant temperature as measured by a 100mm globe thermometer (see IHVE Guide A1-5).	DEG.C
2	R	INFILTRATION	Ventilation rate to external air assuming 25% of walls are external, used in calculating ventilation heat losses. Will be varied if in fact a different proportion of wall is external. If entirely internal half this value is used in calculating infiltration losses to adjoining rooms	AC/HR
3	8A	VENT TYPE	Ventilation type such as: NATURAL CLEANEXT DIRTYEXT WARMINLT INANDEXT AIRCOND DUALDUCT	
10	R	RATE	Required air-change rate	AC/HR
11	8A	VENT TYPE	Repeat values 3-10 if more than one ventilation system required.	

Figure 46. Typical Property Standard for the zone data property *TD.

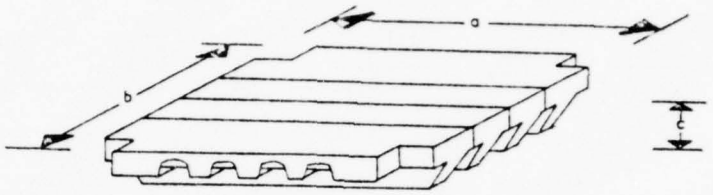
BDS-F1

OXSYS	COMPONENT STANDARD	
Version ONDM	MNEMONIC FLOR:LBAY	CCD 20:7
Date 14 Jan 77		

DESCRIPTION

Pre-cast floor unit bay assemblies for use in bays without intermediate beams.

ORIENTATION

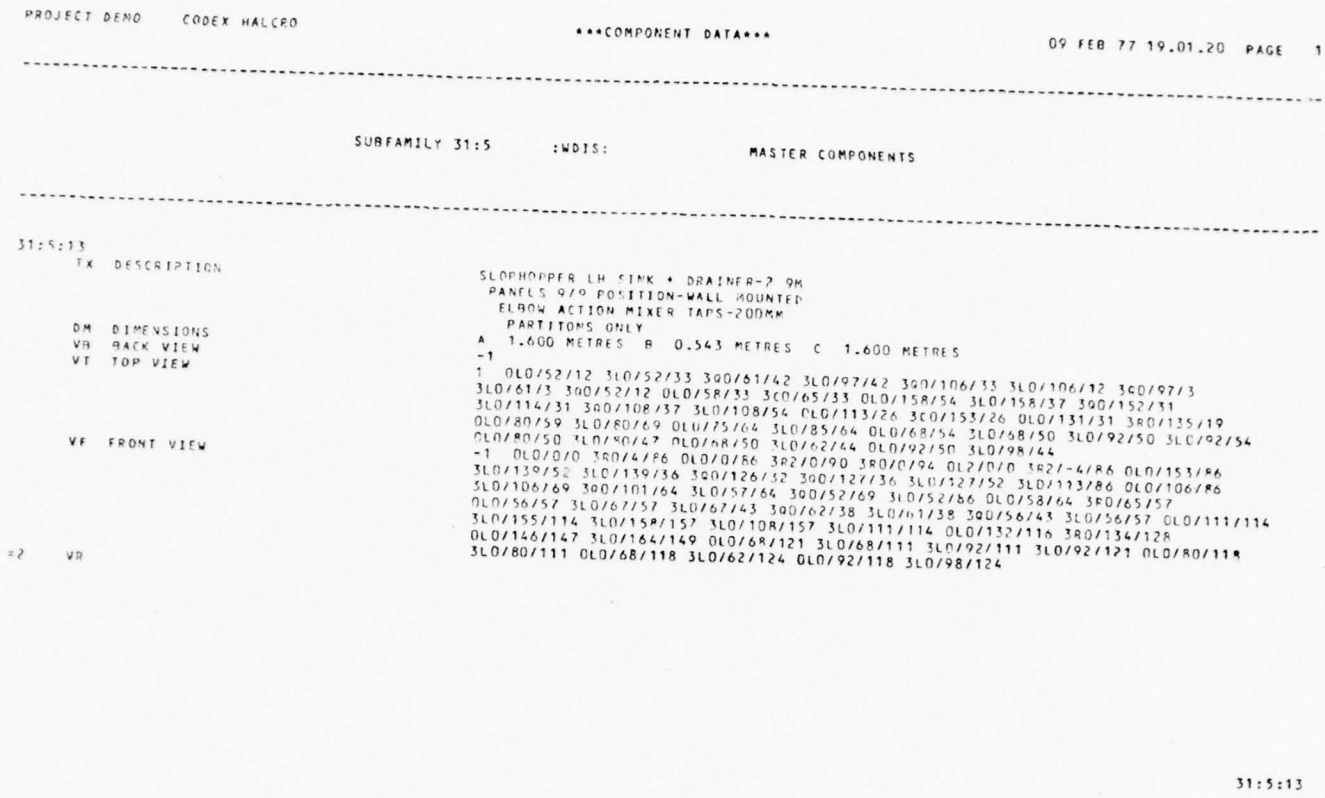


PROPERTIES

CAT	NAME	PM	VAL	REQUIREMENTS
***	Dimensions	DM	1	Length of assembly including fringes of 0.15m (Dimension a).
			2	Width of assembly including fringes of 0.15m (Dimension b).
			3	Depth of assembly, =0.1m (Dimension c).
***	Floor Code	FC		To describe fringe conditions, nominal span of assembly and nominal width of assembly.
***	Component quantities	CQ		To detail the individual components in the assembly.
**	Text	TX		
**	Top view	VT		To indicate individual floor slabs and detail fringes, including curtailment.
**	Front view	VF		To show full depth of ribs.
**	Cross section	VX		To show full depth of ribs.

Figure 47. Component Data Standard.

BDS-F2



```
PROJECT DEMO     CODEX HALCRO                    ***COMPONENT DATA***          09 FEB 77 19.01.20  PAGE   1

                                SUBFAMILY 31:5      :WDIS:          MASTER COMPONENTS

31:5:13
      TX  DESCRIPTION

                                                  SLOPHOPPER LH SINK + DRAINER-2 9M
                                                   PANELS 9/9 POSITION-WALL MOUNTED
                                                    ELBOW ACTION MIXER TAPS-200MM
                                                     PARTITONS ONLY
      DM  DIMENSIONS                              A   1.600 METRES  B  0.543 METRES  C  1.600 METRES
      VB  BACK VIEW                               -1
      VT  TOP VIEW                                1  OL0/52/12 3L0/52/33 3Q0/61/42 3L0/97/42 3Q0/106/33 3L0/106/12 3Q0/97/3
                                                  3L0/61/3 3Q0/52/12 OL0/58/33 3C0/65/33 OL0/158/54 3L0/158/37 3Q0/152/31
                                                  3L0/114/31 3Q0/108/37 3L0/108/54 OL0/113/26 3C0/153/26 OL0/131/31 3R0/135/19
                                                  OL0/80/59 3L0/80/69 OL0/75/64 3L0/85/64 OL0/68/54 3L0/68/50 3L0/92/50 3L0/92/54
      VF  FRONT VIEW                              OL0/80/50 3L0/80/42 OL0/68/50 3L0/62/44 OL0/92/50 3L0/98/44
                                                  -1  OL0/0/0 3R0/4/86 OL0/0/86 3R2/0/90 3R0/0/94 OL2/0/0 3R2/-4/86 OL0/153/86
                                                  3L0/139/52 3L0/139/36 3Q0/126/32 3Q0/127/36 3L0/127/52 3L0/113/86 OL0/106/86
                                                  3L0/106/69 3Q0/101/64 3L0/57/64 3Q0/52/69 3L0/52/86 OL0/58/64 3R0/65/57
                                                  OL0/56/57 3L0/67/57 3L0/67/43 3Q0/62/38 3L0/61/38 3Q0/56/43 3L0/56/57 OL0/111/114
                                                  3L0/155/114 3L0/158/157 3L0/108/157 3L0/111/114 OL0/132/116 3R0/134/128
                                                  OL0/146/147 3L0/164/149 OL0/68/121 3L0/68/111 3L0/92/111 3L0/92/121 OL0/80/118
                                                  3L0/80/111 OL0/68/118 3L0/62/124 OL0/92/118 3L0/98/124
=2    VR

                                                                                                  31:5:13
```

Figure 48. CODEX Data for a component.

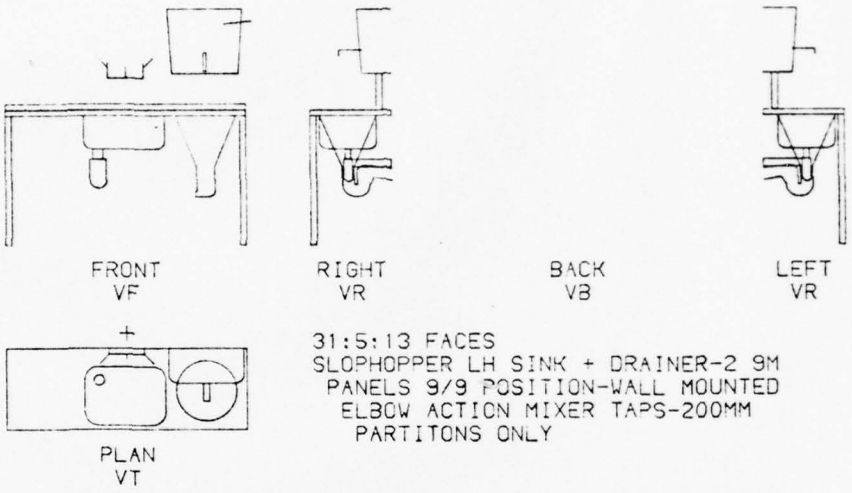


Figure 49. Plot of a catalogued component.

4 OVERVIEW OF AVAILABLE SYSTEMS

4.1 GENERAL CATEGORIZATION

Chapters 2 and 3 provided an overview of the main conceptual and technical issues that should be addressed in implementation of the proposed CAEADS data base system. This chapter surveys the broad range of existing systems that might conceivably be employed by the Corps of Engineers for this purpose. The purpose here is to narrow the field and provide a context for the detailed evaluations and recommendations contained in Chapter 5.

The systems that were surveyed were found to group naturally into the following general categories:

a. General implementation tools

b. Drafting systems

c. Three-dimensional image synthesis systems

d. Surface description systems

e. Polyhedron description systems

f. Network data base systems

g. Polyhedron data base systems

h. Mapping and site description systems

i. Specialized comprehensive architectural/architectural engineering CAD systems

j. Generalized architectural/architectural engineering CAD systems.

4.2 GENERAL IMPLEMENTATION TOOLS

The evolution of general data base implementation tools was described in section 2.2.1. The discussion below is in terms of the classification that was developed there.

4.2.1 Lower Level General-Purpose Programming Languages

Many lower level general-purpose programming languages have

been developed, but FORTRAN and COBOL strongly dominate in their respective fields (ALGOL is not commonly used or widely supported in the United States). With the growth in popularity of minicomputers, BASIC and APL have been increasingly widely used. However, because of the tradition of use of FORTRAN within the Corps of Engineers and the large pool of engineers, architects, and application programmers who know FORTRAN, it seems certain that if a language at this level is to be used, it must be FORTRAN.

4.2.2 Extension of FORTRAN for CAD

The data structuring facilities of FORTRAN are limited to arrays of fixed dimensions. This is entirely inadequate for any kind of sophisticated computer-aided design application. Thus, a number of subroutine packages for extending the data structuring facilities of FORTRAN have been implemented. Typical examples are SLIP[104] and WORM.[105]

SLIP adds in-core lists to FORTRAN. WORM, implemented on a PDP-10 by the Edinburgh Computer Aided Architectural Design group, supports the in-core data structures of the SSHA housing design system. It employs a data structuring concept essentially similar to the plex. However, this type of package only addresses one of the several deficiencies of FORTRAN.

A second deficiency of FORTRAN in most implementations is that it can only read and write sequential files. This problem can be solved by providing a set of subroutine calls from FORTRAN to manipulate structured files on disk, or by enhancing FORTRAN by additional statements for this purpose. In response to this and other needs, a number of general systems for implementing civil engineering CAD systems were developed in the 1960's and early 1970's. These systems typically included some version of FORTRAN extended in this and other directions. Probably the best known and most widely used are ICETRAN, embedded in the ICES system,[106] and GENTRAN, embedded in the GENESYS system.[107] Typically, these types of languages are pre-

[104] J. Weizenbaum, "Symmetric List Processor," Communications of the ACM, Vol 6 (1963).

[105] M. Liardet, WORM: A Data Structuring System for FORTRAN, Edinburgh University Computer Aided Architectural Design (September 1976).

[106] D. Roos (ed.), ICES System General Description, MIT Department of Civil Engineering Report R67-49 (1967).

[107] R. D. Warrender, "GENESYS: A Group of Papers," Computer Languages for Building, CIB Symposium by Correspondence CIB W52, Budapest (1975).

compiled into FORTRAN, and then the FORTRAN is compiled in the normal way.

A more recently developed system in this tradition is IST, with the associated ISTRAN language, developed at the Technische Universitaet Berlin.[108] This was developed initially on Siemens machines, and an implementation for IBM hardware is also being produced.

4.2.3 Generalized Data Base Management Systems

As the technology of handling large, complex disk data bases has developed, the concept of a generalized data base management facility has emerged. There are two basic types of such facilities:

a. Host language systems, which are used to extend the data base management capabilities of languages like FORTRAN

b. Self-contained systems, which aim to handle a range of data base functions in such a way that procedural programming in a language like FORTRAN is not required.

A typical example of a host language system[109] is IMS (IBM Corporation). Some typical self-contained systems[110] are MARK IV (Informatics), TDMS (System Development Corporation), and GIS (IBM Corporation).

Typically, generalized data base management systems provide facilities for

a. Formatting records

b. Defining either hierarchical or network file structures

c. Accessing data

d. Sorting, formatting, tabulating, and generating reports.

The concepts of very high level data definition (DDL) and data manipulation (DML) languages are implemented in many systems.

The major advantage of using a generalized data base management

---

[108] L. H. Klotz, Report on Some International CAD Systems and Activities (October 5, 1976).

[109] CODASYL Systems Committee, Feature Analysis of Generalized Data Base Management Systems (Association for Computing Machinery, May 1971).

[110] CODASYL Systems Committe.

system is that the power of the DDL and DML makes implementing quite large systems quick and simple. However, it is important to note that data base management systems (DBMS) have developed largely in response to the demands of business data processing. Constructs which are appropriate for business applications may not be especially appropriate for implementing computer-aided architectural design systems.

As previously shown, the stored integrated model of a building is likely to have a rather complex structure. Furthermore, it is highly dynamic. Data are continually inserted and deleted in generally highly unpredictable sequences and the fields needed in a record may not be known at the outset (i.e., what attributes are to be used to describe some entity).

Implementing such a model using a data base management system requires the structure of the model (i.e., the entities, attributes, and relations in which it is expressed) to be matched together with facilities for defining entities, attributes, and relations provided by the data base management system. These facilities impose a very definite "view of the world", since the internal routines which process the data base (e.g., in order to get an item) will be limited in number, and will implement particular concepts of structure and access method.

In computer-aided design applications, the "view of the world" imposed by a data base management system is likely to introduce inefficiencies in the following ways:

a. The "natural" organization of the building model may need to be "bent" to fit the data base management system

b. Access via a DML, which takes its cues from a DDL template, introduces additional table look-up operations, and hence overhead

c. Use of fixed format records (which is a common, though not inevitable, feature) leads to wasted space.

These inefficiencies may be quite acceptable in a business environment, where the objective may be rapid implementation of a relatively straightforward information system, on hardware with ample capacity, and by programmers who are not particularly skilled. However, such inefficiencies are very likely to be prohibitive in the highly interactive manipulation of a very complex building description on a small or medium-sized machine. In this latter case, it is better to use a lower level and more general implementation tool, and to employ more highly skilled system designers and programmers to produce a carefully optimized system.

Questions of whether or not acceptable efficiency can be achieved can only be finally decided by detailed analysis, simulation, or benchmarking. Such detailed studies are not within the scope of this project. However, the experience of the Cambridge Computer Aided Design Centre in implementing PDMS (a large piping design system), strongly supports the argument that the advantages to be gained from use of a DBMS to implement a CAD system must be paid for by substantial (and probably unacceptable, in the case of CAEADS) losses in efficiency.

Some relevant examples of applications of the generalized data base management system approach to implementation of CAD and information systems in architecture and related fields are

a. Applications to handling large programmatic data bases for interior space planning[111]

b. The PDMS system for describing and designing large piping complexes, developed at the Cambridge Computer Aided Design Centre[112]

c. The Limerick system, employed by Bechtel Power Corporation[113] for handling engineering construction management data. This has been implemented using Honeywell's IDS data base management software

d. ARIANE, an interactive building products information system maintained by the French Technical Assistance and Documentation Center. The system stores data reference codes which index a microfiche system.[114]

A comprehensive comparison of generalized data base management software cannot be attempted in this study, since there are literally hundreds of different data base management systems in

[111] W. J. Mitchell and J. Hamer, "Space Planning," in J. Gero (ed.), Computer Applications in Architectural Practice (Applied Science, 1977).

[112] R. G. Newell et al., "The Design of Systems for CAD," in J. J. Allan (ed.), CAD Systems (North Holland, 1977); Chemical Engineering Group, PDMS: Technical Information Booklet (Computer Aided Design Centre, Cambridge, England, undated).

[113] R. A. Easton, Overview of the Limerick Data Base Management System (Bechtel Power Corporation, December 1975).

[114] ARIANE Data System, Centre d'Assistance Technique et de Documentation (CATED) (undated).

operation. The best basic references for further details are listed below.[115]

4.2.4 Higher Level Languages

The alternative to employing a language like FORTRAN, suitably extended, is to use one of the available higher level languages which integrally incorporates many of the necessary features.

Many of these are ALGOL-like, for example:

a. PL/1, developed and supported by IBM. Its very extensive features are documented in numerous texts, and need not be described here.

b. ALGOL 68,[116] originally specified by an international committee as a successor to ALGOL 60, and now available in a number of implementations. It has been used successfully in geometric modeling applications (see section 5.2.5).

c. PASCAL,[117] another successor to ALGOL 60, developed in Switzerland. A standard PASCAL has been defined, and there are implementations of various versions.

d. SAIL,[118] developed at Stanford University for artificial intelligence application. It is notable for inclusion of LEAP data structure facilities, and for excellent graphics capabilities.

---

[115] CODASYL Systems Committee, Feature Analysis of Generalized Data Base Management Systems (ACM, 1971); C. J. Date, An Introduction to Database Systems (Addison-Wesley, 1975); R. Ashany and M. Adamowicz, "Readings in Data Base Systems," IBM Systems Journal, No. 3 (1976); and J. Martin, Computer Data Base Organization (Prentice-Hall, 1975).

[116] A. van Wijngaarden et al., Revised Report on the Algorithmic Language ALGOL 68 (Springer-Verlag, Heidelberg, 1976); S. Bourne et al., ALGOL 68 Reference Manual (Computer Laboratory, Cambridge University, 1974).

[117] K. Jensen and N. Wirth, PASCAL: User Manual and Report (Springer-Verlag, New York, 1975).

[118] W. M. Newman and R. F. Sproull, Principles of Interactive Computer Graphics (McGraw-Hill, 1973); K. A. Vanlehn, "SAIL User Manual," Stanford Computer Sciences Reports STAN-CS-73-373 (July 1973).

e. EULER,[119] has been implemented (in an interpreted version) at the University of Utah in a version called EULER-G.[120] This has outstanding graphics capabilities, and has been used for implementation of an experimental building description system.[121]

f. AED,[122] was originally developed in the MIT Computer Aided Design Project in the 1960's, and a version is now marketed as a general system building language by Softech of Waltham, MA. It has very powerful data structuring and input/output (I/O) facilities, making use of the plex concept. Compilers have been developed for IBM 360/370 series machines, the CDC 6000 series, and the UNIVAC 1100 series, and cross-compilers are available to a number of other machines.

In addition, there are some advanced languages specifically intended for use in interactive development of complex models. Among these are

a. L*,[123] a language incorporating some very powerful modeling constructs. A definition is available, and an implementation was produced on the Atlas system at the Cambridge University Computer Laboratory. However, there is no currently available commercial implementation.

b. The Evans and Sutherland Design System,[124] an extensible language, with a syntax based upon the concept of an operand stack. Implementations have been produced for PDP-11 minicomputers, and are commercially available.

---

119 N. Wirth and H. Weber, "EULER: A Generalization of ALGOL, and its Formal Definition," Journal of the ACM, Vol 9, nos. 1 and 2 (January and February 1966).

120 W. M. Newman et al., Programmer's Guide to PDP-10 EULER (University of Utah, Division of Computer Science, June 1970).

121 W. J. Mitchell, "Vitruvius Computatus," in D. Hawkes (ed.) Models and Systems in Architecture and Building (Construction Press, 1975).

122 D. T. Ross, "The AED Approach to Generalized Computer-Aided Design," Proceedings of the ACM National Meeting (1967); D. T. Ross and J. W. Bracket, Automated Engineering Design (AED) Used for Graphics (Softech, undated); and An Introduction to Features and Uses of AED (Softech, 1975).

123 J. C. Gray and J. Tomlinson, L* Programming Guide (Applied Research of Cambridge Ltd., 1974).

124 The E & S Design System (A Brief Preliminary Description) (Evans and Sutherland Computer Corporation, August 10, 1976).

The basic advantages of using any one of these languages is that, in principle, they are much better suited to the task of implementing complex geometric models than extended versions of FORTRAN (some more so than others). Programs can be implemented more rapidly and with fewer errors. Code is likely to be more logically structured, more concise, and more comprehensible. However, there are some important potential disadvantages to consider:

a. Implementations are not available, in most cases, for a wide variety of different types of hardware. In particular, minicomputer implementations often are not available.

b. Support may be difficult to obtain or guarantee. PL/1 is well supported by IBM, and AED and the E & S Design System are supported by their originators, but the rest seem likely to present difficulties.

c. With the exception of PL/1, they are not widely known in the United States.

The best available general sources for further data and descriptions of additional alternatives are listed below.[125]

4.2.5 Comprehensive CAD Implementation Systems

From the earliest days of CAD system development, it was realized that the implementation of almost any fairly ambitious CAD system could be greatly facilitated if certain basic implementation tools were available.[126] This concept has led to the development of a large number of comprehensive CAD implementation systems. Most of these systems feature some or all of the following:

a. A command decoder or problem-oriented language (POL) interpreter

b. A programming language (either FORTRAN, some form of augmented FORTRAN, or a higher level language)

[125] J. Sammet, Programming Languages (Prentice-Hall) (This survey is periodically updated by articles in the Communications of the Association for Computing Machinery, most recently in 1977); and W. M. Newman and R. L. Sproull, Principles of Interactive Computer Graphics (McGraw-Hill, 1973).

[126] D. T. Ross, The AED Approach to Generalized Computer Aided Design, MIT Report ESL-R-305 (MIT, 1967).

c. Sophisticated data structuring and management facilities

d. A graphics system

e. An error-handling security and integrity system

f. An overlay or virtual memory system for programs

g. Some way of handling program modification and recompilation without requiring extensive relinking.

During the 1960's and early 1970's, considerable effort was devoted to the development of implementation systems oriented towards civil engineering applications. Two that have been widely used are ICES,[127] developed at MIT, and used to implement such well-known systems as STRUDL and COGO; and GENESYS,[128] a similar but more recent system developed in Britain. Another similar system, IST (Informationssystem Technik),[129] is currently under development at Berlin Technical University in Germany.

The very large production CAD/CAM systems used by some automobile and aerospace firms also have sets of general implementation facilities at their core. Two of the most important of these large systems are CADANCE,[130] which was developed at the General Motors Technical Center and employs PL/1 in conjunction with Dodd's APL ("apple") ring data structure processor,[131] and MCDONNELL DOUGLAS CAD/CAM,[132] developed by McDonnell Douglas Automation in St. Louis.

Extensive use of CAD now also takes place in ship design. In the United States this has given rise to development of COMRADE,[133] the Computer Aided Design Environment project established at the David W. Taylor Naval Ship Research and Development Center in

---

[127] D. Roos (ed.), ICES System General Description, MIT Department of Civil Engineering Report R67-49 (1967).

[128] R. D. Warrender, "GENESYS: A Group of Papers," Computer Languages for Building CIB Symposium by Correspondence CIB W52, Budapest (1975).

[129] L. H. Klotz, Report on Some International CAD Systems and Acticities (October 5, 1976).

[130] W. Garth, Design Console Technology at General Motors (GM Manufacturing Development, July 1974).

[131] G. Dodd, "APL -- A Language for Associative Data Handling in PL/1," Proceedings of the Fall Joint Computer Conference (1966).

[132] J. L. Lavick, Making Graphics Work, paper presented at the Third Annual Conference on Computer Graphics, Interactive Techniques and Image Processing, University of Pennsylvania (July 1976).

[133] T. R. Rhodes, "The Computer-Aided Design Environment Project (COMRADE)," Proceedings of the National Computer Conference (1973).

support of NAVSEC's Computer Aided Ship Design and Construction (CASDAC) project. COMRADE is one of the most extensive and ambitious systems yet developed. Figures 50 and 51 illustrate its operation schematically.

Three of the most interesting recent examples of new general CAD implementation systems are

a. IDAS,[134] the Integrated Designers' Activity Support System developed at IBM's Tokyo Scientific Center, which is notable for its use of the relational data base concept

b. REGENT,[135] developed at the Nuclear Research Center in Karlsruhe, Germany. REGENT employs an extension of PL/1 called PLR, and is oriented towards interactive graphics use. Essentially, it appears to be an updating of the basic ICES concept.

c. A system developed at the University of Tokyo, by a team headed by M. Hosaka.[136] This system employs an APL-like interactive language called GIL.

Probably the only current implementation system specifically oriented towards highly interactive graphic manipulation of a large and complex data base in a minicomputer environment is OXSYS/BOS.[137] This system extends FORTRAN with a very flexible plex-like data-structuring construct and provides graphics, error handling, security and integrity, command decoding facilities, and a suitable operating system environment. It is designed to avoid, as far as possible, introducing the kinds of inefficiencies which characterize data base management systems (by employing variable format, self-describing records, and by not using a DDL). It has been employed so far to implement several versions of OXSYS, and a polyhedron data base system.

Table 6 compares the basic features of several representative general implementation systems.

[134] H. Matsuka, T. Kawai, and S. Uno, "Integrated Designer's Activity Support System for Architecture," Proceedings of the 1975 Design Automation Conference (1975).

[135] K. Leinemann and E. G. Schlechtendahl, "The REGENT System for CAD," in J. J. Allan (ed.), CAD Systems (North-Holland, 1977).

[136] M. Hosaka et al., "A Software System for Computer Aided Activities," in J. J. Allan (ed.), CAD Systems (North-Holland, 1977).

[137] OXSYS-BOS: A Short Technical Description (Applied Research of Cambridge Ltd., May 1976).

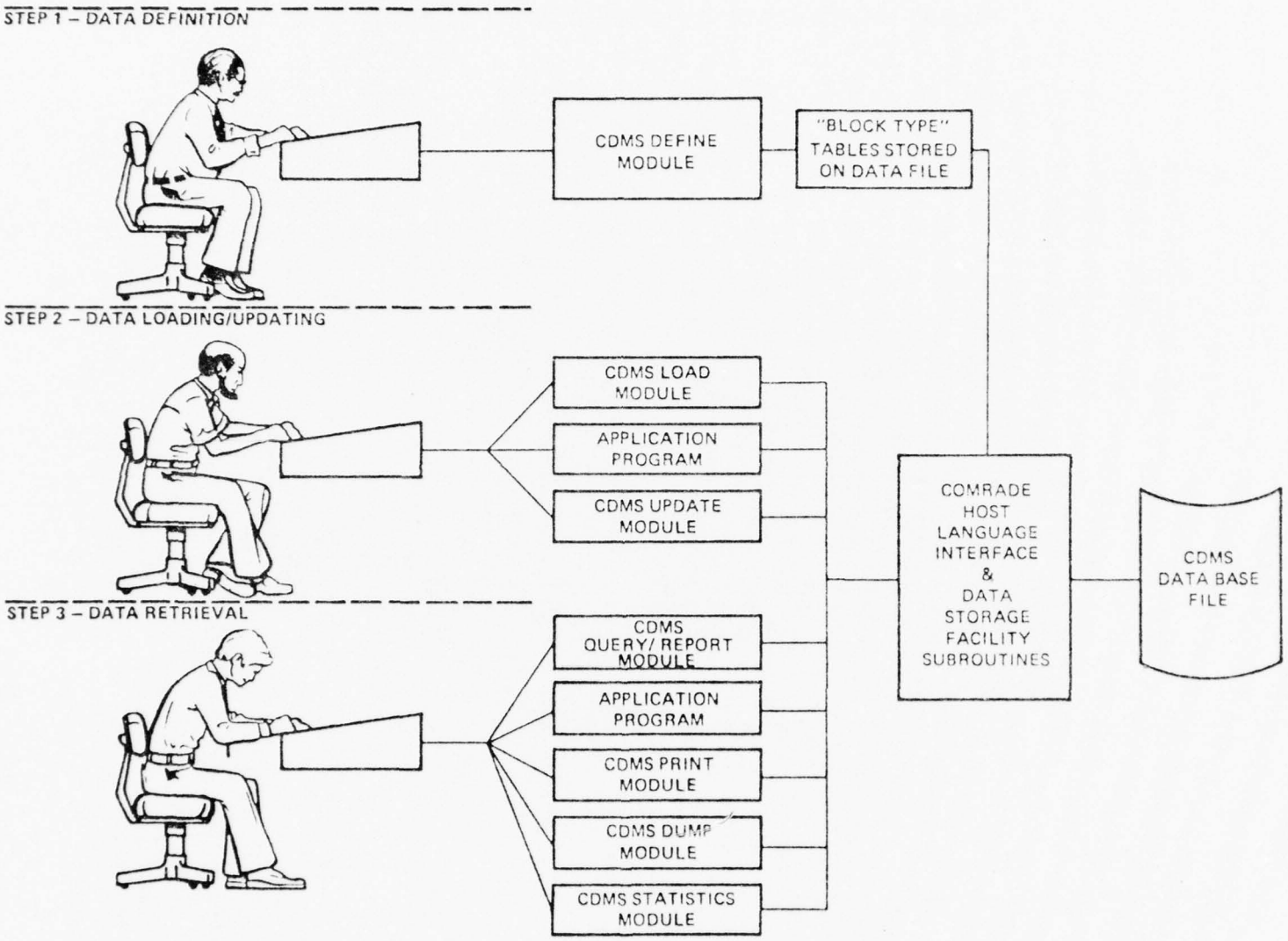


Figure 50. File definition and processing under CDMS.

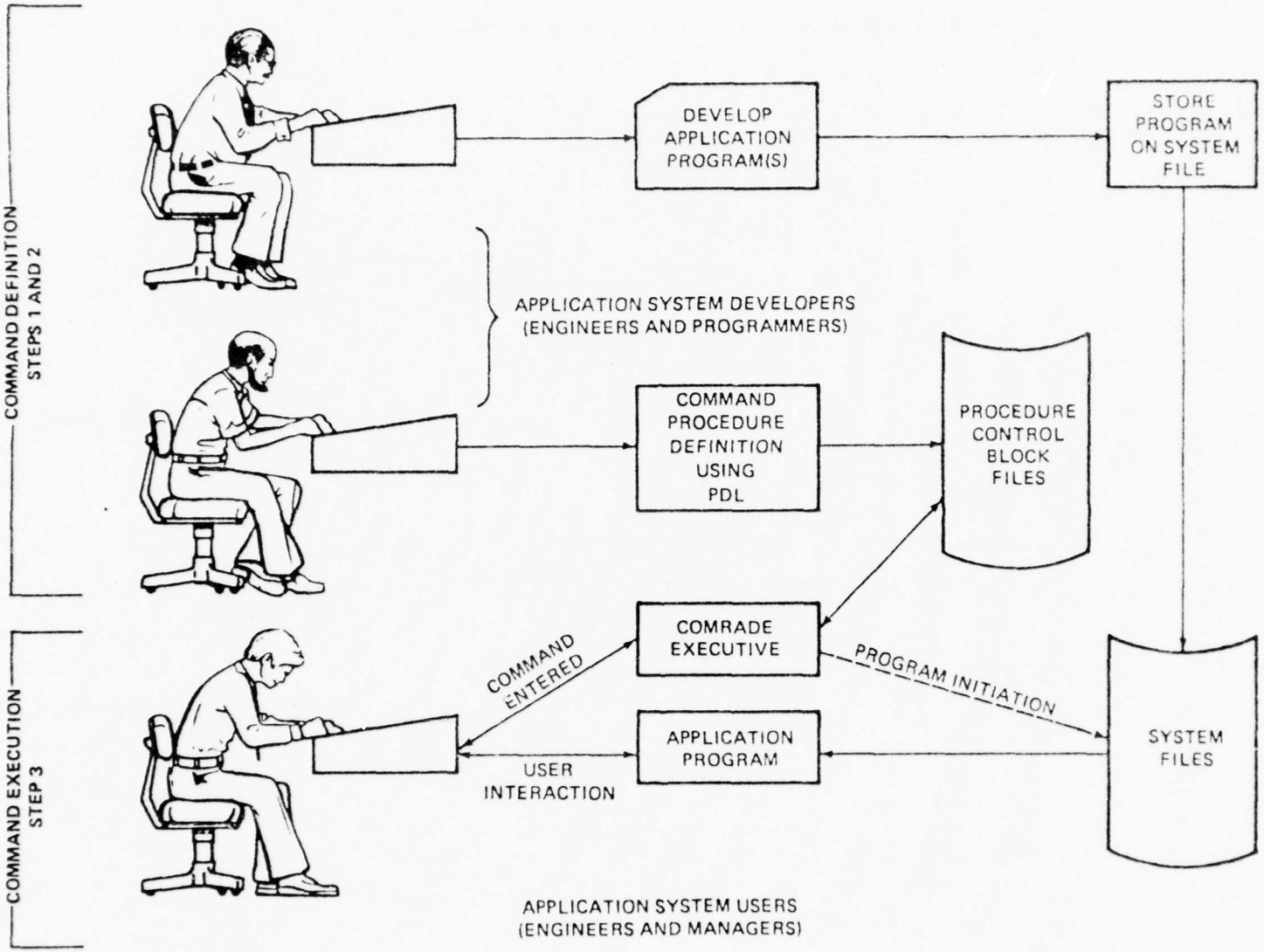


Figure 51. COMRADE command definition and execution phases.

Table 6

Feature Comparison of Some Representative Generalized CAD Implementation Systems

	ICES	GENESYS	IST	REGENT	COMRADE	IDAS	GIL	OXSYS/BOS
ORIGIN	MIT	GENESYS Ltd., Leicester, U.K.	Berlin Technical Univ.	Nuclear Research Center, Karlsruhe, Germany	Naval Ship Engineering Center	IBM Tokyo Scientific Center	Univ.of Tokyo	Applied Research of Cambridge Ltd.
APPLICATION AREA	Civil engineering	Civil engineering	Civil engineering	Civil engineering	Ship design	General	General	General
BASE PROGRAMMING LANGUAGE	FORTRAN	FORTRAN	FORTRAN	PL/1	FORTRAN		GIL, an APL-like interactive language	FORTRAN
EXTENDED PROGRAMMING LANGUAGE	ICETRAN	GENETRAN	ISTRAN	PLR				FORTRAN augmented by data structuring and disk input/output
USER LANGUAGE	POL	POL	POL	POL				Command language
DATA HANDLING FACILITIES	TABLE data base, system. Dynamic arrays	Files of tabular data. Dynamic arrays	Files of tabular data. Dynamic arrays	CODASYL-style DBMS	CODASYL-style DBMS	Relational data base	Arrays, lists, strings	Plex-like data structure concept.
GRAPHICS FACILITIES	No	No	No	Yes	No	Yes	Yes	Yes

4.2.6 Conclusions

The choice of a general implementation tool obviously involves broader considerations within the CAEADS development effort than implementation of the data base. However, data base implementation will make very strong demands upon the general implementation system, and choice of the wrong tool could severely cripple the data base implementation effort.

The task of implementing the CAEADS data base can be characterized as one of highly efficient implementation of a large and complex data base, probably on a fairly small machine. This suggests that fixed-format DDL's, and access via DML's should be avoided, despite the speed and simplicity of implementation that these facilities can provide.

A number of languages which provide flexible and powerful data structuring and disk input/output facilities have been developed, and any one of these might in principle be used to produce an efficient implementation of the proposed data base system with a reasonable amount of programming effort. However, the difficulties which potentially can arise in obtaining and supporting an implementation of an unusual language should be noted.

A second viable alternative is to employ an implementation system which extends FORTRAN in appropriate ways, while avoiding introduction of unacceptable computing overhead.

4.3 DRAFTING SYSTEMS

4.3.1 The Concept of a Drafting System

A computer-based drafting system is essentially intended for manipulation and production of two-dimensional drawings such as maps and engineering and architectural drawings. Although most of these systems internally store the data in two-dimensional coordinate form, some of the more advanced systems have facilities for generating projections and sections from three-dimensional coordinate data. Since the strategy of upward-enhancement of a drafting system into a building description system is sometimes attempted, available drafting systems are discussed in this section.

The basic constructs used in most drafting systems are as follows:[138]

[138] Draft Report of the Drawing Systems Survey for the Property Services Agency (Applied Research of Cambridge Ltd., January 1976).

a. Basic graphics: the computing mechanisms that enable a machine to carry out the simplest drawing functions. They include the drawing of a line from A to B, the moving of the pen to a certain location without drawing a line, the choice of a line in terms of line thickness or line type (dashed, dotted, etc.).

b. Primitives: the mechanism for drawing such simple geometric forms as rectangles, circles, arcs, diagonals.

c. Macros: combinations of basic graphics and primitives, defined as reusable subpictures which can be independently manipulated (for example, rotated and located). The concept of macros is useful as the first level of possible equivalence between building components and their computer graphics. Macros may also be unlocated representations of such building parts as doors, windows, walls, etc., or more abstract definitions, such as a controlling dimensional grid for a building, within which other macros or primitives are located. Macros can be nested within each other so that one subpicture may consist of a specific arrangement of other macros.

d. Layers: a series of subpictures or macros representing different subsets of the total picture information which may be reproduced independently. For example, it may be desirable in mapping to draw contours without roads or vice versa. In other words, the data about roads and contours are classified as different layers and one or both may be selected. The concept of layering picture information can introduce a change of emphasis from simply using the computer to carry out tasks of drawing assemblies of subpictures, primitives, etc., to a situation where data about a building can be classified into different layers and reproduced only as required.

e. Associated data: nongeometric information such as type, cost, or serial number associated with a building element represented by a located instance of macro for the purpose of generating schedules, etc.

4.3.2 Architectural Drafting/Scheduling Systems

Implementing an architectural drafting/scheduling system following these principles is straightforward, and quite a few such systems have been developed. The following is a representative (though certainly not complete) list:

a. Computer Service Inc.,[139] a batch-input system for production of working drawings

b. ARK-2,[140] a minicomputer-based interactive graphics system, commercially marketed as a hardware/software turnkey system. This system includes a set of architectural application programs in addition to drafting facilities, and appears to make some effort to integrate data. It also has a three-dimensional display package, which is discussed in section 4.4.

c. CARBS,[141] a British batch system used for working drawing production, schedule generation; and bills of quantities. This is one of the most ambitious and powerful architectural drafting/scheduling systems currently in production use.

d. Morganelli-Heumann,[142] an efficient minicomputer-based interactive graphics system used by Morganelli-Heumann and Associates of Los Angeles for interior space planning design, drafting, and scheduling.

e. MAN-MAC,[143] an extensive and ambitious interactive system which was originally developed for interior space planning drafting, but which has also been used for architectural and engineering working drawing production. It integrates several additional applications with the graphics data base.

f. Canadian Department of Public Works,[144] a drafting/scheduling system for building design and space management using the GRAPL language developed at Bell Northern Research. This system integrates scheduling and costing applications with the graphics data base.

g. NSW Government Architect,[145] a comprehensive building documentation system intended for use by the New South Wales (Australia) Government Architects' Department.

[139] M. Girardi, "Computer Augmented Drafting System," in W. J. Mitchell (ed.), Proceedings of the EDRA 3 Conference (UCLA, 1972).
[140] ARK-2 (Decision Graphics, undated).
[141] P. Purcell, "Computer Aided Architecture in the United Kingdom," in N. Negroponte (ed.), Computer Aids to Design and Architecture (Petrocelli-Charter, 1975).
[142] W. J. Mitchell and J. Hamer, "Space Planning," in J. Gero (ed.), Computer Applications in Architecture (Applied Science, 1977).
[143] MAN-MAC System Description (S.L.S. Environetics, October 4, 1974).
[144] J. K. Robertson, "A Graphics System for Building Design in the Canadian Government," CAD 76, (IPC Science and Technology Press, 1976).
[145] H. Blue, "Contract Documentation of Dimensionally Coordinated Buildings," DMG-DRS Journal, vol 8, no. 4.

h. TEQUILA,[146] a drafting, scheduling and detailing system for use in design of steel-framed buildings, developed and marketed by the Societé Metallurgique d'Elalia (SOMEL), in France.

4.3.3 General-Purpose Turnkey Drafting Systems

In addition to the specifically architectural drafting/scheduling systems described above, a large number of general-purpose drafting systems are marketed as turnkey hardware/software systems both in America and abroad. Although a particular system may emphasize some special feature, most systems in this category seem roughly comparable, and seem capable of handling architectural drafting/scheduling. They tend to be rather expensive however, and it is noteworthy that they appear not to have penetrated the architectural market significantly.

The following firms marketing drafting systems of this type in the United States were contacted in this study:

a. Applicon, 154 Middlesex Turnpike, Burlington, MA 01803

b. Auto-Trol Corp., 5650 N. Pecos St, Denver, CO 80221

c. Calma, 707 Kifer Road, Sunnyvale, CA 94086

d. Computervision, 201 Burlington Road, Bedford, MA 01730

e. H. Dell Foster, P.O. Box 32581, San Antonio, TX 78216

f. Information Displays Inc., 150 Clearbrook Rd, Elmsford, NY 10523

g. Manufacturing and Consulting Services, Inc., 3195A Airport Loop Drive, Costa Mesa, CA 92626 (AD-2000 System)

h. M & S Computing, P.O. Box 5183, Huntsville, AL 35805

i. Synercom Technology, 6300 Hillcroft, Houston, TX 77036

j. Wang Laboratories, 1 Industrial Ave, Lowell, MA 01851

Accurate details of how these systems handle their data bases are extremely difficult to obtain. However, the available evidence suggests that this type of system would not be adequate for implementation of the CAEADS data base, since their representations of three-dimensional objects are geometrically incomplete, and thus they cannot support the full range of applications.

146L. H. Klotz, Report on Some International CAD Systems and Activities (October 5, 1976).

Table 7

Feature Comparison of Typical Architectural Drafting/ Scheduling Systems

	Computer Service Inc.	ARK-2	CARBS	Morganelli-Heumann	MAN-MAC	Canadian	NSW	TEQUILA
SOURCE	Computer Service Inc., Chicago	Decision Graphics, Southboro, Mass.	CARBS Ltd. Clwyd Co. Council, Wales	Morganelli-Heumann, Los Angeles	SLS, New York	Dept. of Public Works, Ottawa	NSW Govt. Architect, Sydney, Australia	SOMEL France
PRIMARY APPLICATION	Architectural working drawings	General architectural graphics	Working drawings and other construction documents	Interior space planning	Interior space planning	Building design and space management	Working drawings and other construction documents	Steelwork drawings and reports
USER INTERFACE	Batch	Interactive	Batch	Interactive	Interactive	Interactive	Unknown	Unknown

A good test of the geometric completeness of a representation is whether it can support fully automatic sectioning of a three-dimensional object by an arbitrary plane passing through the object. The Computervision and Applicon systems (two of the leaders) require the operator to manually connect up intersection points in correct sequence in order to find the intersection contour.

It should be noted that the authors cannot claim a complete knowledge of the details of the geometric description methods employed in these various systems which are continuously under development. Therefore, comments contained in this study may cease to be accurate at some point in the future. However, acquisition of such a system would be based upon many more considerations than just those relating to the data base. If acquisition were contemplated, the most reasonable strategy would be to invite bids based in part upon a detailed specification which addressed the issues discussed in Chapters 2 and 3 of this report.

4.3.4 Conclusions

As was emphasized in section 3.3.1, a drafting system is not the same thing as a true three-dimensional building description system. Some of the drafting systems described here are very effective within their limits, but the drafting and scheduling applications to which they are well suited cover only a small part of the complete spectrum of building design tasks. In principle, they are not satisfactorily extensible to cover the full spectrum.

A possibility that might be considered, however, is to adapt an existing drafting system to serve as the graphics interface to an integrated three-dimensional building description system.

4.4 THREE-DIMENSIONAL IMAGE SYNTHESIS SYSTEMS

4.4.1 The Three-Dimensional Image Synthesis Task

The synthesis of a perspective or isometric view of a building requires that a form of three-dimensional building description be held in computer memory. However, this type of image synthesis application is a highly specialized task, and image synthesis makes only a subset of the demands on the description that a comprehensive computer-aided design system makes. Furthermore, image synthesis software usually is (appropriately) designed to deal efficiently with these specialized demands, and satisfactory extension to support most other applications is likely to be difficult or impossible.

Among the most important ways in which a three-dimensional building description intended solely for image synthesis is likely to differ from one to be used in computer-aided design are the following:

a. Quantity of data. Even the most spectacular examples of image synthesis, for example Donald Greenberg's well-known simulation of the Cornell University campus, usually only process a few thousand polygons to generate a picture. Elaborate data base management facilities are not normally employed in support of image synthesis.

b. Nongeometric properties. Image synthesis usually requires some facility for assigning a few surface property descriptors to polygons, but not the extensive facilities for handling nongeometric properties required for computer-aided design.

c. Data structure. Most image synthesis systems structure geometric data in a very simple way, e.g., as an unstructured list of polygons. For reasons discussed at length in Chapters 2 and 3, such simple structures do not adequately support CAD.

4.4.2 Comparison of Representative Systems

Surveying all the numerous three-dimensional synthesis systems here would be pointless. However, a good impression of the nature of three-dimensional image synthesis systems can be obtained by considering the following representative examples:

a. GINO,[147] a popular general-purpose two- and three-dimensional graphics package implemented in FORTRAN, which represents objects simply as collections of faces, and produces "wire frame" orthographic or perspective projections.

b. THINGS[148] (THree-dimensional INput of Graphic Solids), developed and marketed by the Computer Aided Design Centre, Cambridge (England). This system generates hidden-line perspectives of objects described in terms of points, lines, and surfaces (up to 750 points, 1200 lines, 500 surfaces). The OXSYS system generates input files to THINGS for perspective production.

c. VIEW,[149] the three-dimensional display package of the

[147] P. A. Woodsford et al., GINO, CAD Group Document (Cambridge University, June 1969).
[148] S. Bensasson, Computer Programs for Building Perspectives (Design Office Consortium, Cambridge, England, 1977).
[149] View 3-D Package (Decision Graphics, undated).

ARK-2 system. This system is quite similar in philosophy to standard two-dimensional drafting systems in its use of the concepts of primitives, macros, and layers.

d. CADDS 3,[150] the three-dimensional graphics software of Computervision's Designer system. Primarily oriented towards drafting of mechanical parts, this system claims considerable sophistication: facilities for associating nongeometric data, extensive graphics options, and a special package for processing parameterized models of parts.

e. Evans and Sutherland Picture System, an extremely powerful graphics processor oriented towards highly interactive manipulation of three-dimensional images. This system is employed to provide a graphics interface for the Evans and Sutherland Design System.

f. PERS,[151] developed by the Computer Unit, School of Architecture and Landscape, Leeds Polytechnic (England). This system is noteworthy because input is in terms of polyhedra, rather than individual points, lines, and faces. Each polyhedron can have up to 16 polygonal faces or 30 edges, and polyhedra can be parameterized.

4.4.3 Conclusions

Image synthesis does not require the same kind of data base support as computer-aided design. Significantly, while two of the best known image synthesis systems (THINGS and the Picture System) have been used as interfaces to computer-aided design systems, they were not used to implement the primary geometric model.

4.5 SURFACE DESCRIPTION SYSTEMS

4.5.1 The Nature of the Surface Description Problem

Both singly and doubly curved surfaces abound on artifacts such as automobiles, aircraft, and even shoes and bottles. A variety of approaches to curved surface description have been deve-

[150] The Designer System, (Computervision Corporation, 1975).
[151] S. Bensasson, Computer Programs for Building Perspectives, (Design Office Consortium, Cambridge, England, 1977).

loped. These can be grouped into the following five broad classes:

a. Point set descriptions, in which the surface is described by a set of points represented by their X, Y, and Z coordinates. These points may be random, in a simple regular array, or in a recursively constructed array. This technique is very commonly used for describing topographic surfaces.

b. Contour line descriptions, which are also commonly used for topographic surface description. The technique can also be extended to description of "streamlined" objects like aircraft and ship forms, and is commonly used for this purpose.

c. Faceted descriptions, in which a curved surface is approximated by polygonal facets (which may or may not be planar). Triangular meshes are most widely used. This technique is commonly used for image synthesis, finite element analysis, and military vulnerability analysis applications.

d. Simple mathematically defined surface descriptions, used where the surface is part of some simply definable object such as a cylinder, sphere, cone, or ellipsoid.

e. Spline curve and surface patch descriptions, based upon the mathematics of Coons patches, Bézier curves, B-splines, etc. These methods are used in CAD systems employed in many branches of manufacturing industry. An up-to-date picture of recent work in this field is given by the conference proceedings listed below.[152]

Curved surface description facilities of CAD systems usually address some or all of the following objectives:

a. To provide a convenient way for designers to parameterize and (via control points) manipulate a surface

b. To produce accurate plotter drawings

c. To produce NC tapes or input for directly computer-controlled NC systems

d. To generate input to finite element analysis programs.

Another major application of curved surface description techniques has been in the aerospace industry, where faceted descriptions of aircraft are used for visual simulation and vulnerability analysis applications.

[152] R. E. Barnhill and R. F. Riesenfeld (eds.), Computer Aided Geometric Design (Academic Press, 1974).

Curved surface description problems of substantial difficulty are rarely encountered in building design. Of course, it can be argued that architects might use curved surface in design more frequently if appropriate description facilities were available to them, but constraints of building construction technology and cost would still impose strict limits. Curved surface description and manipulation facilities are, therefore, not of central importance in architectural CAD. Thus, while most of the curved surface description methods are general enough to allow description of planar-surfaced objects, it would be quite unjustified to base an architectural CAD system on software optimized for dealing with curved surfaces.

The one exception to this rule is the description of topographic surfaces necessary for building site definition. Since this is such a special case, it is best dealt with by a separate topographic surface description subsystem, based upon rather different principles than the building description software.

4.5.2 Curved Surface Description Systems Used in Industry

The following representative examples illustrate the typical characteristics of curved surface description systems used in manufacturing industry:

a. UNISURF,[153] a famous pioneering system initiated by Pierre Bézier in 1962, and used in automobile manufacture at Régie Renault in Paris since 1972. It provides interactive graphics facilities with which car body designers can describe a body shape in terms of surface patches, and NC machinery can be driven from the stored mathematical description.

b. POLYSURF,[154] developed at the Cambridge Computer Aided Design Centre in England. This system is intended for use in interactive design and NC program generation for mechanical components of complex curved form.

c. McDonnell-Douglas CAD/CAM,[155] an extensive system used in

[153] P. Bézier, "Mathematical and Practical Possibilities of UNISURF," in R. E. Barnhill and R. F. Riesenfeld (eds.), Computer Aided Geometric Design (Academic Press, 1974).

[154] A. G. Flutter and R. N. Rolph, "POLYSURF: An Interactive System for the Computer-Aided Design and Manufacture of Components," CAD 76 (IPC Science and Technology Press, 1976).

[155] G. J. Peters, "Interactive Computer Graphics Application of the Parametric Bi-Cubic Surface to Engineering Design Problems," in R. E. Barnhill and R. F. Riesenfeld (eds.), Computer Aided Geometric Design (Academic Press, 1974).

aircraft design and production by McDonnell-Douglas in St. Louis.

d. GM body design,[156] a subsystem of the CADANCE CAD system used by General Motors.

e. FASTGEN,[157] a triangular facet system developed by Falcon Research and Development and employed primarily for military target description and vulnerability analysis. It has powerful data input and verification facilities and a part coding and indexing system, and is interfaced to sophisticated graphics programs.

4.5.3 Topographic Surface Description Systems

The technical problem of topographic surface description has been definitively analyzed by Boehm.[158] Examples of implemented systems are extremely numerous, but the following are representative:

a. Dynamic Graphics of Berkeley California[159] markets an extremely comprehensive and versatile topographic surface description system which accepts data in various forms, performs a number of different transformations, and produces highly sophisticated graphic output. Its basic mode of internal representation is a regular-gridded point set.

b. SSHA Site Layout System,[160] incorporates a ground model input in the form of spot levels, stored as a regular-gridded point set and displayed as contour lines.

c. Applied Research of Cambridge Ltd. markets an interactive graphic system for topographic surface input, transformation, and display. It provides all the transformations between random and gridded data points, triangulated, and contour representations.

[156] W. Garth, Design Console Technology at General Motors (GM Manufacturing Development, July 1974).
[157] T. J. Byrne and J. P. Thompson, Computer Representation of Three-Dimensional Structures (December 3, 1976).
[158] B. W. Boehm, "Tabular Representations of Multivariate Functions, with Applications to Topographic Modeling," Proceedings of the ACM National Meeting (1967).
[159] The Software Development Group, User Manual for the Surface Display Library (Dynamic Graphics Inc., April 1976).
[160] C. Holmes, Ground Modelling in Site Layout, Edinburgh University CAAD Studies (Edinburgh University, undated).

d. GDS,[161] is an ambitious data base system based upon the triangulated surface concept; it is under development at Simon Fraser University.

4.5.4 Conclusions

Curved surface description systems used in industry respond to special requirements which are unlike those of building description. Thus, these types of systems cannot be expected to support architectural applications.

Topographic surface description techniques are well developed, and any one of many available systems might be adapted to serve this function in CAEADS.

4.6 POLYHEDRON DESCRIPTION SYSTEMS

4.6.1 The Nature of the Polyhedron Description Problem

The general importance of polyhedron description problem has been characterized by Braid as follows: "Many common solid objects have shapes characterized by surfaces which are composed of large numbers of simple faces. In mechanical engineering, assemblies and machined components provide examples of the class. They may have hundreds of faces, but most faces will be planar, cylindrical, or some other elementary surface form."[162]

Since the early 1970's, numerous systems have been developed for describing and manipulating descriptions of objects in this class. The major research issues have been techniques for internal representation of shape and algorithms for efficient performance of shape operations, particularly the spatial set operations on polyhedra. (A general and efficient spatial set operation facility is extremely difficult to implement, since a large number of troublesome special cases arise.)

This work on polyhedron description is directly relevant to building description, since most architectural elements (both spaces and solids) are of polyhedral form. Typically, polyhedron description systems cannot be used directly for building descrip-

[161] T. K. Peucker and N. Chrisman, "Cartographic Data Structures," The American Cartographer, Vol 2, No. 1 (April 1975).
[162] I. C. Braid, "The Synthesis of Solids Bounded by Many Faces," Communications of the ACM, Vol 18, No. 4 (April 1975).

tion, because their orientation is towards the design of a single mechanical part or small assemblies of parts. (Polyhedron data base systems, which represent large assemblies of parts, are discussed in section 4.8.) However, the theoretical principles upon which they are based and, in some cases, actual software, can be adapted very readily for use in building description.

4.6.2 Early Systems

Several systems developed during the 1960's are worth brief mention, since they anticipated some of the concepts employed in more recent and advanced systems:

a. Luh and Krolak[163] described an early system for mechanical part description. It could draw a part, compute center of mass and moments of inertia, and generate NC tapes.

b. BE-VISION[164] was an early system which used the concept of describing solids by the Boolean combination of directed surfaces.

c. Comba[165] developed a system which addressed the problem of interference between objects.

d. ARCAID[166] was a system for architectural designs based upon the concept of assembly of parameterized primitive shapes.

e. APT[167] is a widely used programming language for describing objects to be milled on an NC machine. An APT program is a specialized form of procedural representation of an object. The compiler transforms the source program into a tool path sequence. In principle, deriving other geometric data by processing an APT description should be possible, but in practice it would be extremely difficult.

---

[163] Luh and R. J. Krolak, "A Mathematical Model for Mechanical Part Description," Communications of the ACM (February 1965).
[164] R. Weiss, "BE-VISION," Journal of the ACM, Vol 13, No. 2
[165] P. G. Comba, "A Procedure for Detecting Intersections of Three-Dimensional Objects," Journal of the ACM, Vol 15, No. 3
[166] R. M. Wehrli et al., ARCAID: The Architect's Computer Graphics Aid (University of Utah, Department of Architecture, September 1969).
[167] W. H. P. Leslie, Numberical Control User's Handbook (McGraw-Hill, 1970).

4.6.3 Recent Systems

The following recently developed systems, all of considerable sophistication, were identified and investigated:

a. BUILD[168] implemented in FORTRAN and SAL on the Cambridge (England) Computer Laboratory's Titan computer. It provides a vocabulary of six parameterized primitives and an interactive language for composing these primitives into more complex objects by means of the spatial set operations. Internally, it employs a form of boundary representation. This was a very important pioneering piece of work, but it has now been superseded by the following system.

b. GEM[169] implemented in ALGOL 68. It is probably the most advanced polyhedron description system currently available. Both an interpreted command language and compiled procedures written in extended ALGOL 68 can be used for input. The data structure is an extended version of Baumgart's winged-edge polyhedron structure (see GEOMED below). A variety of types of curves and surfaces can be handled, and the system is extensible to handle more. The spatial set algorithms appear to handle all special and degenerate cases satisfactorily.

c. COMPAC,[170] a system recently developed at the Technical University of Berlin.

d. EUCLID,[171] a French system implemented in FORTRAN. A batch version is embedded in FORTRAN, and an interactive version has been implemented in a DEC language called FOCAL. It appears to be oriented primarily towards graphics production, but provides a form of description which could support engineering applications.

e. EUKLID,[172] implemented in Switzerland in the SYMBAL language on a CDC 6500 computer. Although few details of its algorithms have been released, it appears to be very powerful. A batch input ALGOL-

168 I. C. Braid, "The Synthesis of Solids Bounded by Many Faces," Communications of the ACM, Vol 18, No. 4 (April 1975).

169 I. C. Braid, A New Shape Design System, University of Cambridge Computer Aided Design Group Document No. 89 (University of Cambridge, March 1976).

170 G. Spur, J. Gausemeier, and G. Muller, COMPAC: The use of Computer Internal Workpiece Models for Design and Manufacturing, (Technical University of Berlin, 1976).

171 J. M. Brun, EUCLID Manual (LIMSI, Paris, 1976).

172 M. Engeli, "EUKLID - Eine Einfuehrung," (Fides Rechenzentrum, Zuerich, 1974); and M. Engeli, "A Language for 3D Graphics Applications," in A. Gunther et al. (eds.), International Computing Symposium 1973 (North-Holland, 1974).

like language with additional data types for geometric entities is employed by the user to describe objects. It provides a comprehensive spatial set operation facility.

f. GEOMED,[173] implemented in PDP 10 assembly language at Stanford University and intended for use in visual shape recognition research. This system is historically very important because of its introduction of the concepts of Euler operations and of very complete topographical representation clearly separated from the geometric data (the "winged edge polyhedron" data structure).

g. "Grossman,"[174] implemented in PL/1 at IBM Yorktown Heights. Each polyhedron is represented by a PL/1 procedure. This system provides some very useful insights into the concept of procedural geometric modeling. Its major disadvantage is that it reportedly is extremely expensive to run.

h. "Hosaka,"[175] implemented in an extended version of APL (called GIL) at the Institute of Space and Aeronautical Science, Tokyo University. Forms are created interactively using APL-like statements to specify spatial set operations upon directed planes and parameterized primitives.

i. PADL,[176] implemented in FORTRAN at the University of Rochester. This is a research system intended for use in exploring mechanical engineering part-shape description problems. It employs a restricted "built-in" set of parameterized primitives, and provides a language for specifying spatial set operations to construct more complex objects from these primitives.

j. SHAPES,[177] implemented in IBM 360 assembly language at MIT. This system models bodies as Boolean combinations of quadric directed surfaces. Common quadrics like planes, cylinders, and cones are provided by the system, and the user can define others.

---

[173] B. G. Baumgart, Winged Edge Polyhedron Representation, Stanford Artificial Intelligence Memo AIM-179 (Stanford University, October 1972).

[174] D. D. Grossman, "Procedural Representation of Three-Dimensional Objects," IBM Journal of Research and Development (November 1976).

[175] M. Hosaka et al., "A Software System for Computer Aided Activities," in J. J. Allan (ed.), CAD Systems (North-Holland, 1977).

[176] H. Voelcker and A. Requicha, "Geometric Modelling of Mechanical Parts and Processes," Computer Science and Computer Engineering Research Review (University of Rochester, 1976/77).

[177] J. H. Laning et al., SHAPES User's Manual (MIT Draper Laboratory, 1973).

k. TIPS,[178] implemented in FORTRAN at Hokkaido University. Bodies are described and stored internally as Boolean combinations of directed surfaces. An auxiliary point set representation is set up internally for some applications. The orientation is towards NC applications.

l. ANSI[179] method for digital representation of physical object shapes. This is not an implemented computer system, but rather a proposed standard method for communication of shape data in manufacturing industry. It is worth noting here because it incorporates some interesting theoretical innovations which seem likely to influence future polyhedron description systems.

Two outstanding surveys of work on polyhedron description systems have recently been produced.[180] These surveys, together with the results of this investigation, suggest that the list of systems given above was complete as of the time of writing.

4.6.4 Conclusions

Important properties of these 11 systems are compared in Table 8. The published literature on these systems provides a rich source of ideas relevant to the development of the CAEADS data base. Some of the more advanced, transportable software may prove directly usable.

The most important limitations to direct use are likely to be the limitation to a relatively small number of polyhedra in a description; and the lack of availability of a reliable, well-supported implementation. Many of these systems are implemented in obscure languages, or an unusual language, or by research groups that would not be able to provide support.

[178] N. Okino et al., "TIPS-1: Technical Information Processing Systems for Computer Aided Design," Prolamat 73 Proceedings, Budapest (1973).

[179] ANSI Subcommittee Y14.26, Digital Representation of Physical Object Shapes (ANSI, January 8, 1976).

[180] I. C. Braid, Six Systems for Shape Description and Representation: a Review, University of Cambridge Computer Aided Design Group Document No. 87 (University of Cambridge, May 1975); and A. Baer, C. Eastman, and M. Henrion, A Survey of Geometric Modeling, Institute of Physical Planning Research Report No. 66 (Carnegie-Mellon University, March 1977).

Table 8

Feature Comparison of Polyhedron Description Systems

	BUILD	GEM	COMPAC	EUCLID	EUKLID	GEOMED	"GROSSMAN"	"HOSAKA"	PADL	SHAPES	TIPS
SOURCE	University of Cambridge	University of Cambridge	Technical University of Berlin	LIMSI Paris	Fides Rechenzentrum	Standford University	IBM Yorktown Heights	Tokyo University	University of Rochester	MIT Draper Laboratory	Hokkaido University
IMPLEMENTATION LANGUAGE	FORTRAN and SAL	ALGOL 68	Unknown	FORTRAN	SYMBAL	PDP-10 assembly language	PL/1	GIL (extended APL)	FORTRAN	IBM 360 assembly language	FORTRAN
SHAPE MODELING CONCEPT	Boundary representation	Boundary represetantion	Unknown	Boundary representation	Boundary representation	Boundary representation	Boundary representation	Boolean representation	Boolean representation	Boolean representation	Boolean representation
USER INTERFACE	Interactive language	Extension of ALGOL 68	Unknown	Both interactive and batch versions	Algol-like batch input language	(not intended for design applications)	PL/1 procedures	Interactive use of GIL	Interactive language	Routines called by an IBM 360 assembly language program	Unknown

4.7 NETWORK DATA BASE SYSTEMS

4.7.1 Nature of the Spatial Network Description Problem

Many engineering systems may be described as large spatial networks in which the links are essentially homogenous elements. Typical examples are road networks, structural frames, and piping systems. Since the number of elements is likely to be large, a data base management problem may arise when computer modeling of such a system is attempted.

The most common approach to large-scale spatial network description appears to be to employ fairly standard data base management techniques to store descriptions consisting of the following:

a. Network structure (connectivity)

b. Point locations of vertices

c. Parameters describing links (e.g., length, bore, and specification of a pipe)

d. Parameters describing vertices (e.g., type of piping valve or elbow).

4.7.2 Comparison of Representative Systems

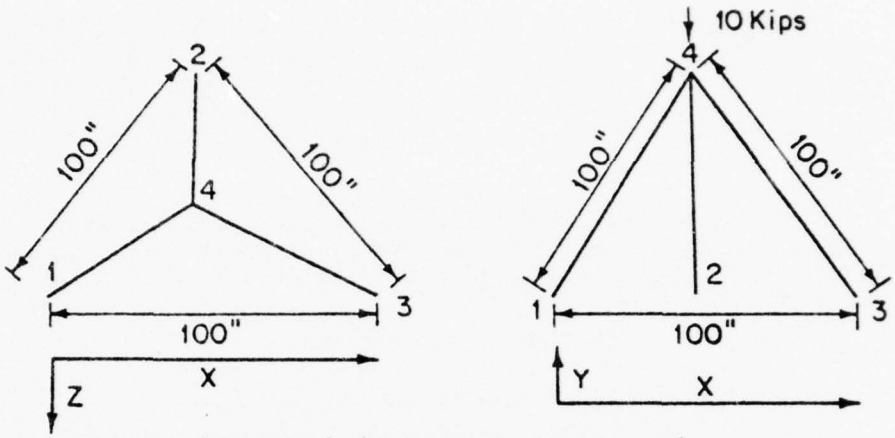
This type of network representation appears to be often implemented within applications programs used by engineering firms. However, due to the proprietary nature of these programs and their frequent lack of documentation, it is difficult to obtain details. The following systems may be taken as representative:

a. STRUDL.[181] Figure 52 illustrates a structural frame description coded in the STRUDL language of ICES. The frame described is a simple steel-tripod made out of 8WF31 rolled steel sections. A description of a larger frame in this format could be maintained using the TABLE data base management subsystem of ICES.

b. PDMS.[182] This system is intended for use in describing very large piping complexes such as chemical process plants. It is implemented on a large minicomputer at the Cambridge (England)

[181] D Roos et al., ICES System General Description, MIT Department of Civil Engineering Report R67-49 (September 1967).

[182] R. G. Newell et al., "The Design of Systems for CAD," in J. J. Allan (ed.), CAD Systems (North-Holland, 1977).




```
STRUDL 'TRIPOD' 'EXAMPLE PROBLEM'
TYPE SPACE FRAME
UNITS INCHES KIPS
JOINT COORDINATES
1 X 0.0  Z 0.0  Y 0.0 SUPPORT
2 Z -86.67  X 50.0 SUPPORT
3 X 100. SUPPORT
4 X 50.  Y 86.67  Z -28.89
MEMBER INCIDENCES
1  1  4
2  2  4
3  3  4
MEMBERS 1, 2, 3 TABLE 'STEELWF' '8WF31'
LOADING 1
JOINT 4 LOAD FORCE Y - 10
LOADING LIST 1
STIFFNESS ANALYSIS
LIST FORCES DISPLACEMENTS ALL
FINISH
```

Figure 52. Typical STRUDL input.

Table 9

Feature Comparison of Network Description Systems

	STRUDL	PDMS	DIME	NETWORKPROCESSOR
APPLICATION	Structural computations	Piping design	Census data management	General
ELEMENTS	Structural members	Pipes	Block faces	Vertices and edges
TYPES OF NETWORKS	Structural frames	Process plants	Street networks	General
IMPLEMENTATION	POL	CODASYL-style DBMS		Interactive system programmed in FORTRAN

Computer Aided Design Centre using a CODASYL-style data base management system. It supports various design development and documentation tasks including automatic pipe routing, interference checking, material scheduling, and production of layout drawings.

c. DIME,[183] a street map file employed by the U.S. Bureau of the Census. The DIME file encodes block faces by storing vertex identifiers for the end points, vertex coordinates (two-dimensional), and codes describing the spaces on either side.

d. Networkprocessor,[184] a generalized system for handling network descriptions, developed at the Technical University of Eindhoven. This system provides a data base facility for maintaining network descriptions, an interactive graphics interface, and a set of graph-theoretic operations for manipulating network descriptions.

Table 9 compares the important properties of these systems.

4.7.3 Conclusions

Although many building subsystems can be represented adequately using the network data base concept for the purpose of particular applications, it is not in itself an adequate foundation for developing a comprehensive building description. The essential reason is that network descriptions take only vertices, edges, and their associated properties and relations as the basic elements of description, whereas support of the full range of architectural applications requires that all types of geometric elements (i.e., vertices, edges, faces, and enclosed volumes) and their respective properties and relations must be represented explicitly or be easily derivable.

4.8 POLYHEDRON DATA BASE SYSTEMS

4.8.1 The Nature of the Polyhedron Data Base Problem

A system like a building or a ship is a large close-packed assembly of polyhedral objects (some solid components, some enclosed

[183] Census Use Study: The Dime Geocoding System (Bureau of the Census, July 1970).

[184] J. Amkreutz and V. E. Tabery, "Een Netwerkprocessor voor Bouwkundige Problemen," Toegepaste Informatica Bouwkunde (Technical University of Eindhoven, 1977).

voids). A complete geometric description can be produced by describing the shape, location, and properties of each of the constituent polyhedra.

To structure this type of description in a useful way, store it compactly, and manipulate it efficiently is an extremely demanding task. The number of elements characteristic of a network representation is combined with the potential complexity of element description dealt with in polyhedron description systems. There is not an absolutely clear-cut distinction between polyhedron description systems and polyhedron data base systems; it is a question of relative capacity. Roughly, polyhedron description systems handle on the order of 100 or fewer polyhedra in a description, while a polyhedron data base system might handle hundreds to tens of thousands.

4.8.2 Rectilinear Systems

To mitigate the difficulty of the implementation problem, implementers of this type of data base have often chosen to impose a restriction that all polyhedra must be rectilinear. The following arguments can be made in support of this course of action:

a. Many buildings are rectilinear, or nearly so, in their geometry. Within particular building systems or methods of construction, a strict rectilinear discipline may be obeyed.

b. For many applications, representation of nonrectilinear components by bounding rectangular parallelepipeds is a sufficiently accurate approximation.

c. Any solid object can be represented to arbitrary accuracy by an assemblage of rectangular parallelepipeds.

Depending upon context, these arguments may or may not provide justification.

The following are among the more important general building description systems based upon rectilinear polyhedra:

a. BUILD,[185] a pioneering system implemented at MIT using the facilities of ICES (not to be confused with Braid's BUILD system).

[185] L. C. Teague, "Network Models of Configurations of Rectangular Parallelepipeds," in G. T. Moore (ed.), Emerging Methods in Environmental Design and Planning (MIT Press, 1970).

b. CADS,[186] another early system developed at UCLA and implemented using the facilities of the Euler language.

c. IMAGE,[187] developed at MIT initially for an interactive space planning application. The description facilities have since been adapted for use in the Corps' SEARCH system.

4.8.3 General Polyhedron Geometry Systems

Polyhedron data base systems not restricted to rectilinear geometry have been a more recent development. However, two systems implemented at Carnegie-Mellon University have shown that it is possible to efficiently handle general polyhedron data bases sufficiently large and well structured to be useful in practical building description, by interactively using minicomputer technology. These systems are

a. BDS,[188] a data base system implemented in the BLISS language on DEC machines. This system has a command language and interactive graphic user interface. Several installations are now in operation.

b. GLIDE,[189] a further development of the concept of BDS which embeds facilities for manipulating the data base in an ALGOL-like programming language. At the time of writing, implementation was nearly complete. GLIDE should make possible the rapid implementation of sophisticated procedural models, automated detailing routines, etc.

A third system is currently under development by Shape Data of Cambridge (England). This FORTRAN system is based upon Braid's GEM polyhedron description system and the OXSYS/BOS implementation tool. It is being developed on a Prime 300 minicomputer, however, the current stage of development is still too early for technical details to be available.

---

[186] W. J. Mitchell, "Vitruvius Computatus," in D. Hawkes (ed.), Models and Systems in Architecture and Building (The Construction Press, 1975).

[187] G. Weinzapfel and S. Handel, "IMAGE: Computer Assistant for Architectural Design," in C. Eastman (ed.), Spatial Synthesis in Computer Aided Building Design (Wiley, 1975).

[188] C. Eastman and J. Lividini, "A Database for Designing Large Physical Systems," Proceedings of the National Computer Conference (1975).

[189] C. Eastman and M. Henrion, Language for a Design Information System, Institute of Physical Planning Research Report (Carnegie-Mellon University, 1976).

Table 10 compares the important properties of these polyhedron data base systems.

4.8.4 Conclusions

The polyhedron data base approach to comprehensive building description appears to be correct in principle and has been shown (by the implementation and application of actual systems) to be feasible in practice. The more highly developed systems of this type, i.e., BDS and GLIDE, deserve very serious consideration for use in implementing the CAEADS data base.

4.9 HIGH LEVEL BUILDING DESCRIPTION SYSTEMS

4.9.1 The Nature of the High Level Building Description Problem

A high level building description system (see Chapter 2 for a discussion of the concept of level in this sense) can be developed using a polyhedron data base of some kind as the foundation. A high level system makes extensive use of very high level input operations such as automated sizing, location, detailing, and building assembly. It may exploit specific knowledge about a particular construction method or building type in order to provide sophisticated consistency-checking facilities, and it almost certainly has associated specialized application programs.

4.9.2 Examples of High Level Systems

All the currently implemented high level systems have been developed for use in Britain by public sector organizations concerned with particular systems or methods of construction. They are the following:

a. CEDAR 2,[190] a system developed in England as a collaborative effort between the Department of the Environment and the Royal College of Art. It provided facilities for design of rectilinear steelframed buildings in the South Eastern Architects' Consortium (SEAC) component system, and was implemented in FORTRAN on an ATLAS

[190] P. A. Purcell, "Computer-Aided Architecture in the United Kingdom," in N. Negroponte (ed.), Computer Aids to Design and Architecture (Petrocelli-Charter, 1975).

Table 10

Feature Comparison of Polyhedron Data Base Systems

	BUILD (Teague)	CADS	IMAGE	BDS	GLIDE
IMPLEMENTATION LANGUAGE	ICETRAN plus TABLE in ICES	Euler	Unknown	BLISS	BLISS
GEOMETRY HANDLED	Rectangular parallele-pipeds	Rectangular parallele-pipeds	Rectangular parallele-pipeds	Planar ploy-hedra	Planar polyhedra
SIZE OF DATA BASE HANDLED	Small	Small	Small	Large	Large
USER INTERFACE	Command language	Command language	Command language	Command language	Programming language
SOURCE	MIT	UCLA	MIT	Carnegie-Mellon	Carnegie-Mellon

computer. It has now been superseded by the following system.

b. CEDAR 3,[191] a system developed by the British Property Services Agency (PSA) for use with the PSA Method of Building, a rather loosely defined "open" building system. The major orientation is towards supporting environmental analysis, preliminary mechanical services design, and economic evaluation at the sketch design stage for office buildings. It is implemented in FORTRAN on a PDP-10, and has a storage tube graphics terminal interface. It assumes rectilinear geometry. Some high level input operations and built in analysis routines are provided. A high degree of portability is claimed, and marketing of the system is apparently intended.

c. HARNESS[192] is a highly specialized system specifically intended for use in design of Harness hospitals. These hospitals are of strictly modular, highly standardized design, and are constructed using component building systems. Because it operates in this limited context, the HARNESS computer system provides a much higher level of design automation than any other system to date. It is conceptually interesting as an illustration of the potential power of very high level design operations, but it is of no practical use outside the Harness hospital program. Development was carried out by Applied Research of Cambridge Ltd. for the British Department of Health and Social Security.

d. OXSYS 1, a system developed by Applied Research of Cambridge Ltd. on an ATLAS computer for use in hospital design using the Oxford Method component building system. This has now been superseded by the more general system implemented on a Prime minicomputer, which is described below. The early version is noted only because references to it appear in some early published literature, and the two versions should not be confused. All references to OXSYS in this survey refer to the newer, generalized system.

---

191 D. Charlesworth and G. Webster, CEDAR 3: The Philosophy of Basic Software for Computer Aided Design, CEDAR report D12/ep/CAD76 (Property Services Agency, 1975); D. Charlesworth, Spatial Organization, CEDAR report D15/tq/SPATORG (Property Services Agency, 1975); D. Charlesworth, Non-Geometric Data: a Structural Scheme, CEDAR report D15/gn/ATTFIL (Property Services Agency, 1975); and CEDAR 3 (Property Services Agency, 1977).

192 J. Jacobsberg, "Computer Design Aids for Large Modular Buildings," in D. Hawkes (ed.), Models and Systems in Architecture and Building (The Construction Press, 1975).

e. OXSYS BDS and DDS level systems.[193] The OXSYS/BDS system is a generalized facility for description of buildings of essentially rectilinear geometry. Several installations are in use. Its facilities can also be used for very rapid implementation of specialized high level systems (DDS). Each one of these DDS systems is used for design within some particular, essentially rectilinear method of construction. So far, several of these systems have been implemented and put into use by different organizations and more are projected. They can provide powerful automated sizing selection, location, detailing, and building assembly operations. Implementation is in FORTRAN plus the facilities provided by the BDS and BOS levels of OXSYS.

f. SSHA[194] is a system specifically intended for use in design of single-family houses. It was developed by the Edinburgh University Computer Aided Architectural Design Group, and has been used successfully in practice for some time by the Scottish Special Housing Association. It is programmed in FORTRAN, is quite portable, and is distributed commercially by Applied Research of Cambridge Ltd. Rectangular floor plans and pitched roofs are assumed. A variety of standard methods of construction can be handled.

Table 11 compares the important properties of these systems.

Mitchell[195] provides further discussions of these systems, together with examples of various types of output and illustrations of projects developed with their aid.

4.9.3 Conclusions

Because high level systems of this type are inherently specialized to some degree, no such system could be expected to support the full range of the Corps' activities. However, OXSYS/BDS appears to be general enough to deal with a substantial subset of the Corps' buildings.

[193] E. M. Hoskins, Integrated Computer Aided Building and the OXSYS Project (Applied Research of Cambridge Ltd., June 1976).

[194] A. Bijl et al., ARU Research Project A25/SSHA-DOE: House Design, Edinburgh University Computer Aided Architectural Design Studies (Edinburgh University, 1971).

[195] W. J. Mitchell, Computer Aided Architectural Design (Petrocelli-Charter, 1977).

Table 11

Feature Comparison of High Level Building Description Systems

	CEDAR 3	HARNESS	OXSYS	SSHA
Source	PSA, London	ARC (for U.K. Dept. of Health and Social Security)	ARC	Edinburgh University CAAD
Construction context	PSA method of building	Component systems	General	Various options
Building type	Medium rise offices etc.	Harness hospitals	General	Small single family housing
Geometry handled	Recti-linear	Strictly modular and recti-linear	Recti-linear	Recti-linear
Implementation language	FORTRAN	FORTRAN	FORTRAN plus BOS	FORTRAN plus data structure package

4.10 SITE DESCRIPTION SYSTEMS

4.10.1 The Nature of the Site Description Problem

As noted previously, the site description problem differs from the building description problem because it involves description of curved topographic surfaces and surface features rather than assemblages of polyhedra. (Description of topographic surfaces was discussed in section 4.5.) This section focuses on description of surface features such as roads, paving, property lines, and utility networks (which may be subsurface).

Automated mapping of surface features and utility networks at urban, regional, and even larger scales is now a substantial industry. In addition, a certain amount of software intended specifically for building site feature description is available.

4.10.2 Large-Scale Cartographic Data Base Systems

In recent years, a number of large-scale cartographic data base systems have been implemented in the United States. Typical features of these systems are

a. Efficient data input and editing techniques

b. Geographic coordinate indexing

c. Heavy emphasis upon graphics, either interactive displays or plotted maps.

Since the scale of these systems tends to be very large, they are often implemented as custom-developed hardware/software packages. The following are representative:

a. Brooklyn Union Gas Co.[196] maintains a data base with an interactive graphics interface describing their 3,600-mile (5 760 km) pipe network.

b. Texas Highway Department[197] has developed an extensive system for topographic mapping and highway design. This system has found application nationwide.

[196] K. A. Godfrey, "Making Maps by Computer," Civil Engineering-ASCE (February 1977).
[197] K. A. Godfrey, "Making Maps by Computer."

c. CMIMS Complete Map Information Management System[198] is an ambitious system proposed for Sacramento County, CA. The scope of this system encompasses all the types of base maps required by the county.

d. LUMIS[199] is a comprehensive land use management information system. A pilot version for an area of Southern California has been implemented by Jet Propulsion Laboratory.

Many of the drafting systems discussed in section 4.3 are suitable for this type of application and often can be supplied with cartographic application software.

Good surveys of some of the basic technical issues involved in development of data structures and algorithms to support these types of systems are listed below.[200]

4.10.3 Building Site Description Systems

Building site surface feature and utility network description involves some additional issues that are not addressed by the types of systems described so far. Among the more important of these issues are

a. Provision of high level and very high level input operations for use in designing and describing site plans,

b. Compatibility with and relation to the three-dimensional building description.

Thus, in principle it is probably more satisfactory to develop software specifically for building site description than to attempt to adapt generalized cartographic data base software. Some examples of building site description software systems which respond to these issues are described in the following subsections.

---

198 V. W. Cartwright and J. P. Alessandri, "Computer Cartography Offers County Unlimited Combinations and Considerable Savings," Civil Engineering - ASCE (November 1976).

199 C. Paul, LUMIS System: Final Report (Jet Propulsion Laboratory, 1976).

200 R. Baxter, Computer and Statistical Techniques for Planners (Methuen, 1976); and T. K. Peucker and N. Chrisman, "Cartographic Data Structures," The American Cartographer, Vol 2, No. 1 (April 1975).

4.10.4 High Level Input Operations

Two examples of high level languages for input of site plan descriptions are

a. SIPLAN,[201] developed by Yessios at Carnegie-Mellon University. Facilities are provided for defining generalized "patterns" of circulation elements and polygonal objects related in particular ways, for description of site shapes, and for automated layout of a particular "pattern" within a particular site shape.

b. A language developed by Arnold,[202] at the University of Cambridge, for input of descriptions of housing site layouts.

This provides a simple notation of mnemonics for various features, with associated parameters.

4.10.5 Relation of Site and Building Elements

At least four different levels of integration can be distinguished between the topographic surface description, site feature description, and the building description:

a. Systems in which no attempt is made to represent relations between various different elements, as exemplified by simple drafting systems.

b. Systems which relate site elements (such as roads, paved surfaces, etc.) to the topographic surface model, but not to each other. Examples of this type of system include the site plan subsystem of the CARBS drafting/scheduling system (see section 4.3.2); and a site plan system recently developed at Imperial College, London.[203]

c. Systems which relate site elements both to the topographic surface model and to each other. A good example of this type of system is the site layout system employed by the SSHA for housing projects.[204] This is separate from, but compatible with,

[201] C. Yessios, "Formal Languages for Site Planning," in C. Eastman, (ed.), Spatial Synthesis in Computer Aided Building Design (Wiley, 1975).

[202] D. Arnold, A Computer Model of Housing Layout, unpublished Ph. D. dissertation (University of Cambridge, 1976).

[203] Computer Aided Design, vol 8, No. 4 (October 1976).

[204] A. Bijl and G. Shawcross, "Housing Site Layout System," Computer Aided Design, Vol 7, No. 1 (January 1975).

the housing design system discussed previously. It has extensive input and output facilities and allows quite detailed and accurate description of topography, building outlines, foundations, roads and paths, landscape and car parking areas, fences and retaining walls, and drainage networks. Various analysis routines, including detailed costing, are incorporated. The system is implemented in FORTRAN, and appears to be easily transportable. The system was developed by the Edinburgh University Computer Aided Architectural Design group.

d. Systems which aim at full integration between topographic surface description, site elements, and the three-dimensional building description. This involves some quite subtle issues relating to description of the interface between building and ground. The site description subsystem of OXSYS[205] (which has not yet been fully implemented) aims at full integration. The SSHA house design and site layout system are also interfaced together, so that location-dependent data for each instance of a house-type or a site (e.g., quantities and costs for foundations and drainage) can be generated.

4.10.6 Conclusions

The principles of site description seem well understood, and much software is available. The main implementation task would be to integrate the various desirable facilities into one coherent system, and to achieve proper integration with the three-dimensional building description.

---

[205] P. Richens, OXSYS-BDS User's Manual (Applied Research of Cambridge Ltd., February 1977).

5 EVALUATION AND SELECTION OF SOFTWARE

5.1 SUMMARY OF CRITERIA

This section summarizes criteria to be used in either selecting existing software for CAEADS data base implementation or evaluating the design of proposed new software.

5.1.1 General Structure of the System

The following general system features should all be considered essential:

a. Levels, modularity, and extensibility. The software should be implemented as a multilevel, modular, extensible system, as discussed in section 2.2.

b. Integration. The concept of a single, integrated data base from which reports and specially formatted files are generated as needed should be the basis of system design (see sections 2.1.5 and 2.3.9).

c. Types of files. Provision should be made for the following types of files (or some scheme that can be shown to be equivalent):

(1) Building project file
(2) Project catalogue file
(3) General reference catalogue files
(4) Site file.

Appropriate distinctions should be drawn between definitive data, working data, and historical data. (See section 2.3 for further discussion.)

d. Support of entire design process. The system should support the entire design system process from project inception to detailed documentation without discontinuities (see section 2.3).

e. Procedural modeling. To reduce redundancy, allow compact storage, facilitate consistency checking, and assist implementation of very high level input operations, convenient and powerful facilities for procedural expression of relations must be provided (see section 2.4 for further discussion).

f. Security and integrity. Adequate facilities for maintaining data security and integrity must be provided (see section 2.5 for further discussion).

g. Extensibility. It is likely that a system like CAEADS will almost continuously be extended and modified to support new applications. The building description system should recognize and support this.

5.1.2 The Geometric Model

The following properties of the geometric model maintained by the system should all be considered essential:

a. Geometric completeness. The internal building model should be geometrically complete (not a collection of two-dimensional projections), as discussed in section 3.1.

b. Accuracy. The model should represent shapes and dimensions with sufficient accuracy to support all normal architectural and architectural engineering applications (see sections 3.1.5 and 3.1.6).

c. Nongeometric properties. Provision should be made for the assignment of arbitrary and unpredictable numbers and type of nongeometric properties to geometric entities (see section 3.1.7).

d. Association. Provision should be made for a user to associate arbitrary collections of elements into sets and networks as described in section 3.1.8.

e. Access. Provision should be made for efficient spatial access of elements, for naming and classifying both spaces and physical elements in ways that are useful in design, and for accessing elements via these classification systems (see section 3.2).

f. Lower level shape input operations. Convenient lower level shape input operations are required, in particular an extensible library of parameterized shape primitives and the spatial set operations (see section 3.3).

g. Higher level input operations. Provision should be made for convenient and rapid implementation of higher level input operations, such as automated selection, dimensioning, location, detailing, and building assembly (see section 3.3.5).

h. Output. Both literal and diagrammatic graphics, a full range of projections, and sectioning should be supported (see section 3.4).

i. Consistency checking and maintenance. Minimally, spatial

conflict testing should be supported. Additional consistency checking and maintenance facilities are highly desirable (see section 2.4.4).

j. Documentation systems. An efficient and convenient documentation system should be available (see section 3.5.4).

5.1.3 Computing Resources and System Performance

Experience with the implementation of OXSYS, CEDAR, and BDS has demonstrated that this type of three-dimensional building description system should not require massive computing resources, and that high performance interactive graphics operation is not an unreasonable expectation. The following computing resource and system performance criteria are suggested:

a. Low-cost, widely available computing environment. Implementation should be feasible within a low-cost, widely available type of computing environment, e.g., medium size minicomputer with disk, tablet, and storage tube graphic terminal facilities.

b. Adequate capacity. The system should have the capacity to represent large and complex buildings. The capacity to describe a large, well-serviced general hospital on a single disk-pack would be a reasonable capacity criterion.

c. Efficient handling of small projects. Conversely, the system should not be too cumbersome or impose too many overheads to allow efficient handling of small projects. A single family house would be a reasonable benchmark building for testing this.

d. Real-time interaction. A designer must be able to manipulate the design in real time via a convenient interactive graphics interface.

e. Reliability and robustness. An interactive system is useless in practice if it crashes too frequently, or if it is plagued by bugs which disrupt a designer's work on the system.

5.1.4 Acquisition Considerations

In addition to meeting the technical criteria summarized above, any software acquired must meet certain additional practical criteria. These are general and well known, but for completeness they are restated here:

a. Stability. There must be evidence that the software has been adequately tested and debugged for use in a production environment.

b. Support. Either it must be feasible for the user organization to accept full responsibility for support, or there must be evidence that the supplier is capable of maintaining adequate support for an acceptable period. Support involves at least software maintenance, user assistance, and consultation on extensions and modifications.

c. Documentation must be available, complete, and of high quality.

d. Availability. It must be asked whether and under what circumstances and contractual arrangements the software would be available to the Corps, what adaption would be required and how long this would take, and what the cost would be.

5.1.5 Compatibility With Existing Corps Software

Since the Corps has already invested considerable effort in software development and user training, the compatibility of the data base system with existing software is an important issue. *The following criteria should be applied:*

a. Interfacing existing application programs to the system should be possible without too much difficulty. Minimally, it should be possible to write simple programs to extract needed input data from the integrated data base and format it for input to the application programs.

b. Retraining of FORTRAN application programmers for other languages should be avoided if possible.

5.1.6 *Weighting of Criteria*

These various criteria might be assigned relative weightings in rather different ways, depending upon

a. The ways in which the costs and benefits of the overall integrated CAEADS system are to be defined and evaluated.

b. The precise role assigned to the three-dimensional building description systems within the integrated CAEADS system.

(This question had not been settled definitively at the time of writing this report.)

c. The relative importance attached to relatively rapid implementation of a minimal working system versus spending more time and implementing a more sophisticated system.

In the discussions of possible implementation strategies given in the next section, an effort is made to clearly indicate the weighting assumptions which underlie the recommendations.

5.2 ANALYSIS OF SOFTWARE WITH RESPECT TO THE SUMMARIZED CRITERIA

Brief documentation on OXSYS, the Evans and Sutherland System, and GLIDE is provided by the publications listed below.[206] For convenient comparative analyses of the features of a range of typical, widely-used data base management systems, reference should be made to the CODASYL study and the text by Date.[207]

Table 12 compares these systems with respect to the criteria which were summarized in section 5.1.

[206] P. Richens, OXSYS-BDS Users Manual (Applied Research of Cambridge Ltd., February 1977); OXSYS-BOS: A Short Technical Description (Applied Research of Cambridge Ltd., May 1976); The E & S Design System (A Brief Preliminary Description) (Evans and Sutherland Computer Corporation, August 10, 1976); C. Eastman and J. Lividini, "Database for Designing Large Physical Systems," Proceedings of the National Computer Conference (1975); C. Eastman, Preliminary User's Manual for BDS 10 Institute of Physical Planning, (Carnegie-Mellon University, September 1976); and C. Eastman and M. Henrion, Language for a Design Information System, Institute of Physical Planning (Carnegie-Mellon University, 1976).

[207] CODASYL Systems Committee, Feature Analysis of Generalized Data Base Management Systems (Association for Computing Machinery, May 1971); and C. J. Date, An Introduction to Database Systems (Addison-Wesley, 1976).

Table 12

Comparison of Alternative Systems Against Summarized Criteria

	GENERALIZED IMPLEMENTATION TOOLS			POLYHEDRON DATABASE SYSTEM	HIGH LEVEL BUILDING DESCRIPTION SYSTEM
CRITERIA	DBMS	E & S	OXSYS/BOS	GLIDE	OXSYS/BDS
GENERAL STRUCTURE OF THE SYSTEM					
1. Levels and modularity	Provides many of the necessary lower level facilities to implement a system meeting these criteria	Provides many of the necessary lower level facilities to implement a system meeting these criteria	Provides many of the necessary lower level facilities to implement a system meeting these criteria	Designed to facilitate implementation of modular, extensible multilevel systems	Structured as a modular, extensible three-level system
2. Integration				Implements the concept of an integrated data base	Implements the concept of an integrated data base
3. Types of files				Makes the basic distinction between project description and catalogue files. Provides facilities to implement all the necessary types of files	Has project description and catalogue files, and makes distinction between working and definitive data
4. Support of entire design process				Potentially yes	Yes
5. Procedural modeling	Basically a different philosophy	Extremely powerful facility for this purpose		Extremely powerful facility for this purpose	In a limited way
6. Security and integrity	Facilities usually provided	No specific security facilities provided	Facilities provided	No specific security facilities provided	Facilities provided
7. Extensibility	Facilitates extension and modification of higher level systems	Facilitates extension and modification of higher level systems	Facilitates extension and modification of higher level systems	Facilitates extension and modification of higher level systems	Designed for easy implementation of additional applications

Table 12 (Cont'd)

CRITERIA	DBMS	E & S	OXSYS/BOS	GLIDE	OXSYS/BDS
THE GEOMETRIC MODEL					
1. Geometric completeness	Provides many of the lower level facilities to implement a system meeting these criteria	Provides many of the lower level facilities to implement a system meeting these criteria	Provides many of the lower level facilities to implement a system meeting these criteria	Full three-dimensional description	Full three-dimensional description
2. Accuracy				Shape accuracy possible to any desired level	Shape accuracy limited for nonrectilinear buildings
3. Nongeometric properties				Facilities provided	Facilities provided
4. Association				Facilities provided	Facilities provided
5. Access				Well developed spatial access, and facilities for implementing other access paths as desired	Well developed spatial access and other access paths useful in building design
6. Lower level shape input operations		Extensibility and syntax of language greatly facilitate the implementation of needed geometric input operations	Provides command decoding system which would facilitate implementation of these facilities	Creation of parameterized primitives Dimensioning and locating these primitives Efficient spatial set operations	Digitization Dimensioning and locating rectangular parallelepipeds
7. Higher level input operations				Provides powerful facilities for rapid implementation of higher level operations	Extensively provided in BDS level systems: automated selection, location, sizing, detailing, and building assembly
8. Output	Usually provide powerful facilities for printed report generation, but no specific graphics facilities	Interfaced to E&S Picture System, making very high quality refreshed graphics possible. Specially formatted files for application programs are produced by special access programs in the language	Provides basic interactive and plotter graphics	Refreshed graphics interface Sections to arbitrary planes Specially formatted files for application programs are produced by special access programs in GLIDE	Developed storage tube graphics and plotter interfaces. Interface to THINGS for perspectives Sections possible in directions parallel to axes only Complete facility at BDS level for automated pro-

Table 12 (Cont'd)

CRITERIA	DBMS	E & S	OXSYS/BOS	GLIDE	OXSYS/BDS
THE GEOMETRIC MODEL					
8. Output (cont'd)					duction of working drawings Specially formatted files for application programs are produced by special FORTRAN access programs
9. Consistency checking and maintenance	Some advanced systems provide generalized facilities which could be of assistance	No specific facilities	No specific facilities	Designed to facilitate implementation of consistency checking and maintenance	Spatial consistency checking and maintenance facilities provided
10. Documentation		Self documentation of commands			Self documentation of catalogues
COMPUTING RESOURCES AND SYSTEM PERFORMANCE					
1. Computing environment	Implementations of various systems available for a wide variety of environments	Currently implemented for all models of the PDP-11 minicomputer Transfer to other machines could require considerable effort	Currently implemented on a Prime minicomputer A portability study has been conducted This indicates that implementation of BOS on other computers would take variable amounts of time up to a maximum of 1 man-year, depending upon the type of machine A FORTRAN compiler is required	GLIDE is implemented in BLISS, which apparently only runs on DEC machines Implementation for a PDP-10 should be completed by summer 1977, and for a PDP-11 a few months later	Requires an OXSYS/BOS implementation

Table 12 (Cont'd)

CRITERIA	DBMS	E & S	OXSYS/BOS	GLIDE	OXSYS/BDS
COMPUTING RESOURCES AND SYSTEM PERFORMANCE					
2. Adequate capacity	Normally designed to handle data bases of the scale required	Software addresses a very large virtual memory It is very difficult to evaluate the amount of memory that would actually be needed for a building description Seems likely to have adequate capacity	Designed to handle data bases of the scale required	It is expected that a larger building may have 250,000 to 500,000 parts, and that GLIDE would require 40 to 50 words to describe each part. Thus it should be possible to describe large buildings on a single disk-pack	Designed to handle large, highly serviced general hospitals. Thus it is possible to describe all but the very largest Corps buildings on a single disk-pack
3. Efficient handling of small projects	Depends upon type of higher level system implemented	Depends upon type of higher level system implemented	Depends upon type of higher level system implemented	Anticipated yes	Yes
4. Real time interaction	Likely to be a problem	Depends upon type of higher level system implemented Designed to support interactive use	Designed specifically for high efficiency in supporting interactive CAD systems	Attention has been given to optimizing data base operations Compilation of subroutines should allow efficient execution Likely to provide good response, but this will not be known for certain until implementation is complete and tested	Demonstrated to be very fast for most common operations
5. Reliability and robustness	Usually good	No data	Demonstrated to be highly reliable in development and production environments	No data available until implementation complete	Performing with high reliability in use in production
ACQUISITION CONSIDERATIONS					
1. Stability	Many commercially released systems available	Commercially released system, in use in the field	Commercially released system, in use in the field	Research-oriented system, currently in final stages of implementation Being implemented by a highly competent team following sound principles	Commercially released system, in use in the field

Table 12 (Cont'd)

CRITERIA	DBMS	E & S	OXSYS/BOS	GLIDE	OXSYS/BDS
ACQUISITION CONSIDERATIONS					
2. Support	Many of the available systems are very well supported	Supported by Evans and Sutherland Computer Corporation, a leader in the field of computer graphics	Supported by Applied Research of Cambridge Ltd., a firm established in Britain and North America, which specializes in architectural and urban planning software	Produced by a university-based research group, and currently not supported by a commercial organization Prospects for continued active support from the research group are not certain, but some may be available	Supported by Applied Research of Cambridge Ltd.
3. Documentation	Good documentation available for most systems	Application programmer documentation available	Very extensive technical documentation and programming manual developed	Language description available	Complete user manual available
4. Availability	Many systems marketed in comptetition	Commercially available from Evans and Sutherland Computer Corporation Sells for $ 40,000 for an unrestricted licence Contact: John Warnock	Commercially available from Applied Research of Cambridge, Inc. Single version sells for around $ 55,000 Contact: Mary Oliverson	Will be available from Institute of Physical Planning, Carnegie-Mellon University Since development was supported with Federal research funds, will be available to the Corps at direct cost of transfer (perhaps $ 1,000) Will be accessible for experimental purposes, via ARPANET Developers are actively interested in seeing the system applied in practice Contact: Charles Eastman	Commercially available from Applied Research of Cambridge, Inc. Cost depends on specific options chosen Single version of complete system sells for around $ 150,000 (including BOS) Extension to deal with pitched roofs and to provide interfaces to existing Corps programs would take about 6 man-months Contact: Mary Oliverson
COMPATIBILITY WITH EXISTING CORPS SOFTWARE					

Table 12 (Cont'd)

CRITERIA	DBMS	E & S	OXSYS/BOS	GLIDE	OXSYS/BDS
COMPATIBILITY WITH EXISTING CORPS SOFTWARE					
1. FORTRAN	Some systems provide FORTRAN-compatible DML's	No	System is implemented in FORTRAN throughout, with the exception of a few low-level routines	No	System is implemented in FORTRAN throughout
2. Interfacing existing application programs	High level report generation facilities might be used to produce input files in appropriate format for existing programs	Routines for generating appropriately formatted input files for existing programs could be written in the language	Routines for generating appropriately formatted input files for existing programs could be written in FORTRAN	Routines for generating appropriately formatted input files for existing programs could be written in GLIDE	Routines for generating appropriately formatted input files for existing programs could be written in FORTRAN

5.3 APPROACH TO COMPARATIVE EVALUATION

The analysis shows that all the alternatives have both strengths and weaknesses. However, none of the weaknesses appear to be necessarily fatal, and an implementation strategy based upon any of these systems could potentially meet the CAEADS requirements.

To provide further data upon which to base a choice, it would be very useful to carry out some kind of comparative analysis on OXSYS, E & S, and GLIDE, and one or two representative data base management systems.

Formalized benchmarking in the normal sense is not possible, since the three systems are not directly comparable. They are at different levels, and would make different kinds of contributions to implementation of the CAEADS data base. Furthermore, in each case, the amount of effort needed to create a building description depends very heavily upon how closely the currently available library of elements and procedures matches the characteristics of the particular building to be described.

The recommended approach, which would be feasible and relatively inexpensive, would be to have an architect experienced in computer-aided design create, with the cooperation of the developers of the systems, a nontrivial building description using each one of these systems. He/she should keep a critical record of his/her experiences, paying specific attention to

a. Ease or difficulty of learning the system

b. Ease or difficulty of creating the description using the facilities of the system

c. Specific limitations and freedoms that seem important

d. Response and reliability

e. Computing resources consumed.

Records of terminal sessions and copies of graphic output produced from the description should be presented.

Approximately 8 to 12 man-weeks of work would be needed to perform this investigation reasonably thoroughly. Some travel costs would also be involved, since current implementations of the systems are in widely scattered locations (Cambridge, England; Pittsburgh, PA; and Palo Alto, CA).

A more ambitious type of comparative study, suggested by Charles Eastman,[208] would be to let small development contracts to a number of development groups to develop kernel systems. Either way, the objective of obtaining further detailed data on the most promising systems would be served.

[208] C. Eastman, Report of the CAEADS Review Meeting (May 11-12, 1977).

6 CONCLUSIONS AND RECOMMENDATIONS

6.1 CONCLUSIONS

The main conclusions of this study are as follows:

a. Implementation of a data base system for comprehensive three-dimensional building description in CAEADS will be a complex task, requiring sophisticated techniques; however, it is feasible. It is the correct approach to system integration, and the benefits from its use should be major.

b. No currently available software exactly matches the criteria for three-dimensional building description software to be used in CAEADS, but several high quality systems which would be of great assistance in implementing the needed facilities are available. These are OXSYS, the Evans and Sutherland Design System, and GLIDE. The use of a data base management system (DBMS) may be feasible but is not recommended.

6.2 RECOMMENDATIONS

The following recommendations are made based upon these conclusions:

a. The Corps should take steps to acquire the necessary software and commence the necessary development contracts in order to implement the CAEADS data base. This should be given high priority, since the character and success of the entire CAEADS system will be very strongly influenced by the properties of the data base system.

b. If possible, the proposed comparative analysis (section 5.4) of OXSYS, the Evans and Sutherland Design System, GLIDE, and DBMS systems should be undertaken at the first available opportunity.

c. If the proposed comparative analysis is not conducted, the currently most highly recommended strategy is a staged strategy based upon OXSYS, as described in section 5.5.

d. Strategies based upon the Evans and Sutherland Design System and upon GLIDE (see sections 5.2 and 5.3) are also recommended for serious consideration.

e. The possibility of using a DBMS should be noted, but it is not highly recommended (see section 5.2.4).

REFERENCES

Abstracts of Computer Programs (Automated Procedures for Engineering Consultants (APEC), 1972)

American National Standards Institute, Subcommittee Y14.26, "Digital Representation of Physical Object Shapes," American National Technical Report (January 8, 1976)

J. Amkreutz and V. E. Tabery, "Een Netwerkprocessor voor Bouwkundige Problemen," Toegepaste Informatica Bouwkunde (Technical University of Eindhoven, 1977)

A. Appel and P. M. Will, "Determining the Three-Dimensional Convex Hull of a Polyhedron," IBM Journal of Research and Development (November 1976)

ARIANE Data System, Centre d'Assistance Technique et de Documentation (CATED) (undated)

ARK-2 (Decision Graphics, undated)

D. Arnold, A Computer Model of Housing Layout, unpublished Ph. D. dissertation (University of Cambridge, 1976)

R. Ashany and M. Adamowicz, "Readings in Data Base Systems," IBM Systems Journal, No. 3 (1976)

A. Baer and C. Eastman, The Consistency of Integrated Databases for Computer Aided Design, Institute of Physical Planning Research Report (Carnegie-Mellon University, 1976)

A. Baer, C. Eastman, and M. Henrion, A Survey of Geometric Modeling, Institute of Physical Planning Research Report No. 66 (Carnegie-Mellon University, March 1977)

R. E. Barnhill and R. F. Riesenfeld, Computer Aided Geometric Design (Academic Press, 1974)

B. G. Baumgart, Winged Edge Polyhedron Representation, Stanford Artificial Intelligence Memo AIM-179 (Stanford University, October 1972)

R. Baxter, Computer and Statistical Techniques for Planners (Methuen, 1976)

S. Bensasson, Computer Programs for Building Perspectives (Design Office Consortium, Cambridge, England, 1977)

P. Bézier, "Mathematical and Practical Possibilities of UNISURF," in R. E. Barnhill and R. F. Riesenfeld (eds.), Computer Aided Geometric Design (Academic Press, 1974)

A. Bijl et al., ARU Research Project A25/SSHA-DOE: House Design, Edinburgh University Computer Aided Architectural Design Studies (Edinburgh University, 1971)

A. Bijl et al., SSHA-DOE Site Layout Project, Phase 1 Final Report, CAAD Studies (Edinburgh University, January 1974)

A. Bijl and G. Shawcross, "Housing Site Layout System," Computer Aided Design, Vol 7, No. 1 (January 1975)

H. Blue, "Contract Documentation of Dimensionally Coordinated Buildings," DMG-DRS Journal, Vol 8, No. 4

B. W. Boehm, "Tabular Representations of Multivariate Functions, with Applications to Topographic Modeling," Proceedings of the ACM National Meeting (1967)

S. Bourne et al., ALGOL 68 Reference Manual (Computer Laboratory, Cambridge University, 1974)

I. C. Braid, A New Shape Design System, University of Cambridge Computer Aided Design Group Document No. 89 (University of Cambridge, March 1976)

I. C. Braid, Six Systems for Shape Description and Representation: a Review, University of Cambridge Computer Aided Design Group Document No. 87 (University of Cambridge, May 1975)

I. C. Braid, "The Synthesis of Solids Bounded by Many Faces," Communications of the ACM, Vol 18, No. 4 (April 1975)

J. Brainin, Functional Description for the Integrated Ship Design System-(ISDS) Report 4663 (Naval Ship Research and Development Center, April 1975)

J. M. Brun, EUCLID Manual (LIMSI, Paris, 1976)

T. J. Byrne and J. P. Thompson, Computer Representation of Three-Dimensional Structures (December 3, 1977)

V. W. Cartwright and J. P. Alessandri, "Computer Cartography Offers County Unlimited Combinations and Considerable Savings," Civil Engineering - ASCE (November 1976)

CEDAR 3 (Property Services Agency, London, 1977)

Census Use Study: The Dime Geocoding System (Bureau of the Census, July 1970)

J. Chalmers, "The Development of CEDAR," International Conference on Computers in Architecture (British Computer Society, 1972)

J. Chalmers, P. Sampson, and G. J. Webster, "Data Structures Used in CEDAR," in M. A. Sabin (ed.), Programming Techniques in Computer Aided Design (NCC Publications, 1974)

D. J. Charlesworth, Spatial Organization: A Set-Based Method of Representing Relationships Between Spaces (Property Services Agency, London, 1976)

Chemical Engineering Group, PDMS: Technical Information Booklet (Computer Aided Design Centre, Cambridge, England, undated)

D. L. Childs, Description of a Set-Theoretic Data Structure, CONCOMP Technical Report No. 3 (University of Michigan, 1968)

CODASYL Systems Committee, Feature Analysis of Generalized Data Base Management Systems (Association for Computing Machinery, May 1971)

E. F. Codd, "A Relational Model of Data for Large Shared Data Banks," Communications of the ACM, Vol 13, No. 6(June 1970)

P. G. Comba, "A Procedure for Detecting Intersections of Three-Dimensional Objects," Journal of the ACM, Vol 15, No. 3

Computer Aided Design, Vol 8, No. 4 (October 1976)

Data Base Task Group (DBTG) of CODASYL Programming Language Committee Report (April 1971)

C. J. Date, An Introduction to Database Systems (Addison-Wesely, 1975) Chapters 6 & 20

The Designer System (Computervision Corporation, 1975)

Draft Report of the Drawing Systems Survey for the Property Services Agency (Applied Research of Cambridge Ltd., January 1976)

G. Dodd, "APL -- A Language for Associative Data Handling in PL/1," Proceedings of the Fall Joint Computer Conference (1966)

The E & S Design System (A Brief Preliminary Description) (Evans and Sutherland Computer Corporation, August 10, 1976)

C. Eastman, Assessment of Work on CAEADS (February 2, 1977)

C. Eastman, "Databases for Physical System Design" A Survey of U.S. Efforts," CAD 76 Proceedings (IPC Science and Technology Press, 1976)

C. Eastman, Feasibility and a Proposed Development of AEADS II (April 1976)

C. Eastman, "General Purpose Building Description Systems," Computer Aided Design, Vol 8, No. 1 (January 1976)

C. Eastman, "An Interrogation Language for Building Descriptions," in D. Hawkes (ed.), Models and Systems in Architecture and Buildings (The Construction Press, 1975)

C. Eastman, Preliminary User's Manual for BDS 10, Institute of Physical Planning (Carnegie-Mellon University, September 1976)

C. Eastman, Report of the CAEADS Review Meeting (May 11-12, 1977)

C. Eastman, "Representations for Space Planning," Communications of the ACM, Vol 13, No. 4 (April 1970)

C. Eastman and M. Henrion, Language for a Design Information System, Institute of Physical Planning Research Report (Carnegie-Mellon University, 1976)

C. Eastman and J. Lividini, "A Database for Designing Large Physical Systems," Proceedings of the National Computer Conference (1975)

C. Eastman and J. Lividini, Spatial Search, Institute of Physical Planning Research Report No. 55 (Carnegie-Mellon University, May 1975)

R. A. Easton, Overview of the Limerick Data Base Management System (Bechtel Power Corporation, December 1975)

M. Engeli, EUKLID - Eine Einfuehrung (Fides Rechenzentrum, Zuerich, 1974)

M. Engeli, "A Language for 3D Graphics Applications," in A. Gunther et al. (eds.), International Computing Symposium 1973 (North-Holland, 1974)

A. G. Flutter and R. N. Rolph, "POLYSURF: An Interactive System for the Computer-Aided Design and Manufacture of Components," CAD 76 (IPC Science and Technology Press, 1976)

W. Garth, Design Console Technology at General Motors (GM Manufacturing Development, July 1974)

K. A. Godfrey, "Making Maps by Computer," Civil Engineering-ASCE (February 1977)

M. Girardi, "Computer Augmented Drafting Systems," in W. J. Mitchell (ed.), Proceedings of the EDRA 3 Conference (UCLA, 1972)

D. P. Greenberg, "Computer Graphics in Architecture," Scientific American (May 1974).

J. C. Gray and J. Tomlinson, L* Programming Guide (Applied Research of Cambridge Ltd., 1974)

D. D. Grossman, "Procedural Representation of Three-Dimensional Objects," IBM Journal of Research and Development (November 1976)

J. A. Hamilton, A Survey of Data Structures for Interactive Graphics, Rand Corporation Memorandum RM-6145-ARPA (April 1970)

G. N. Harper, "BOP: An Approach to Building Optimization," Proceedings of the ACM National Conference (1968) pp 575-83.

D. Hawkes and R. Stibbs, The Environmental Evaluation of Buildings Working Papers 15, 27, 28 (University of Cambridge Centre for Land Use and Built Form Studies, 1970)

C. Herot, "Graphical Input Through Machine Recognition of Sketches," Computer Graphics, Vol 10, No. 2 (1976)

L. J. Hoffman, "Computers and Privacy: A Survey," Computing Surveys, Vol 1, No. 2 (June 1969)

C. Holmes, Ground Modelling in Site Layout, Edinburgh University CAAD Studies (Edinburgh University, undated)

M. Hosaka et al., "A Software System for Computer Aided Activities," in J. J. Allan (ed.), CAD Systems (North-Holland, 1977)

E. M. Hoskins, Integrated Computer-Aided Building and the OXSYS Project (Applied Research of Cambridge Ltd., June 1976)

R. J. Hubbold, "Multi-Level Data Structures, Segmentation and Paging," in M. A. Sabin (ed.), Programming Techniques in Computer Aided Design (NCC Publications, 1974)

An Introduction to Features and Uses of AED (Softech, 1975)

J. Jacobsberg, "Computer Design Aids for Large Modular Buildings," in D. Hawkes (ed.), Models and Systems in Architecture and Building (The Construction Press, 1975)

K. Jensen and N. Wirth, PASCAL: User Manual and Report (Springer-Verlag, New York, 1975)

L. H. Klotz, CAEADS - Critique and Recommendations (February 24, 1977)

L. H. Klotz, Report on Some International CAD Systems and Activities (October 5, 1976)

A. N. Kolmogorov, "Logical Basis for Information Theory and Probability Theory," IEEE Transactions on Information Theory, Vol IT-14, No. 5, (September 1968) pp 662-664

G. Lafue, Recognition of Three Dimensional Objects from Orthographic Views, Institute of Physical Planning Research Report (Carnegie-Mellon University, 1976)

J. H. Laning et al., SHAPES User's Manual (MIT Draper Laboratory, 1973)

J. L. Lavick, Making Graphics Work, paper presented at the Third Annual Conference on Computer Graphics, Interactive Techniques and Image Processing, University of Pennsylvania (July 1976)

K. Leinemann and E. G. Schlechtendahl, "The REGENT System for CAD," in J. J. Allan (ed.), CAD Systems (North-Holland, 1977)

W. H. P. Leslie, Numerical Control User's Handbook (McGraw-Hill, 1970)

M. Liardet, WORM: A Data Structuring System for FORTRAN, Edinburgh University Computer Aided Design (September 1976)

Luh and R. J. Krolak, "A Mathematical Model for Mechanical Part Description," Communications of the ACM (February 1965)

J. Martin, Computer Data Base Organization (Prentice-Hall, 1975)

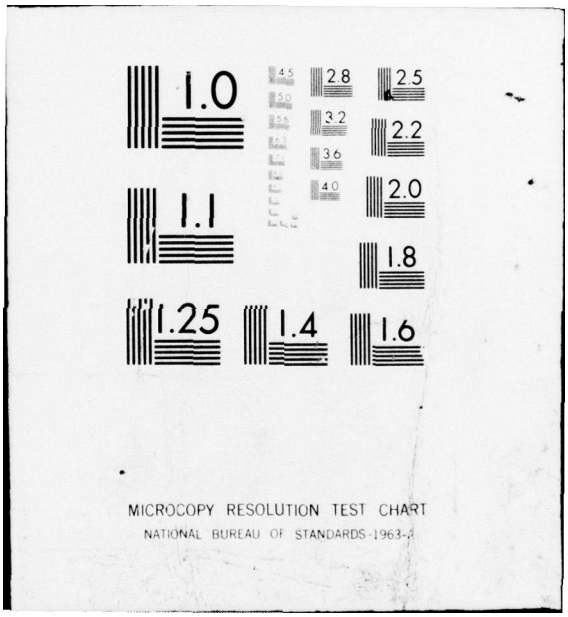
McGraw-Hill Information Systems Company Sweets Catalog Vols 1 - 3 (McGraw-Hill, 1976)

MAN-MAC System Description (S.L.S. Environetics, October 4, 1974)

D-A052 040 APPLIED RESEARCH OF CAMBRIDGE LTD (CANADA) F/G 13/13
COMPUTER REPRESENTATION OF THREE-DIMENSIONAL STRUCTURES FOR CAE--ETC(U)
FEB 78 W J MITCHELL, M OLIVERSON DACA88-77-C-0001
NCLASSIFIED CERL-TR-P-86 NL

3 OF 3
AD
A052 040

END
DATE
FILMED
5-78
DDC


1.0
1.1
1.25
1.4
1.6
1.8
2.0
2.2
2.5
2.8
3.2
3.6
4.0
4.5
5.0
5.6
MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

L. March, "Boolean Description of a Class of Built Forms," in L. March (ed.), The Architecture of Form (Cambridge University Press, 1976)

H. Matsuka, T. Kawai, and S. Uno, "Integrated Designer's Activity Support System for Architecture," Proceedings of the 1975 Design Automation Conference (1975)

T. Maver, "PACE 1: Program for Building Appraisal," Architects Journal (The Architectural Press, April 23, 1973)

W. J. Mitchell, Computer Aided Architectural Design (Petrocelli-Charter, 1977)

W. J. Mitchell, "Vitruvius Computatus," in D. Hawkes (ed.) Models and Systems in Architecture and Building (The Construction Press, 1975)

W. J. Mitchell and J. Hamer, "Space Planning," in J. Gero (ed.), Computer Applications in Architecture (Applied Science, 1977)

W. J. Mitchell, J. P. Steadman, and R. S. Liggett, "Synthesis and Optimization of Small Rectangular Floor Plans," Environment and Planning B, Vol 3, No. 1 (1976)

M. G. Moore, D. M. Brotton, and F. Glover, "Beam Details for Steel Framed Buildings," CAD 76 Proceedings (IPC Science and Technology Press, 1976)

N. Negroponte, The Architecture Machine (MIT Press, 1970)

N. Negroponte, "Recent Advances in Sketch Recognition," Proceedings of the 1973 AFIPS Conference (1973)

A. Newell, "Artificial Intelligence and the Concept of Mind," in R. C. Schank and K. M. Colby (eds.), Computer Models of Thought and Language (W. H. Freeman, 1973)

M. E. Newell, "Man Machine Communication in Three Dimensions," in R. E. Barnhill and R. F. Riesenfeld (eds.), Computer Aided Geometric Design (Academic Press, 1974)

M. E. Newell and D. C. Evans, "Modeling by Computer," in J. J. Allan (ed.), CAD Systems (North-Holland, 1977)

R. G. Newell et al., "The Design of Systems for CAD," in J. J. Allan (ed.), CAD Systems (North-Holland, 1977)

W. M. Newman et al., Programmer's Guide to PDP-10 EULER (University of Utah, Division of Computer Science, June 1970)

W. M. Newman and R. F. Sproull, Principles of Interactive Computer Graphics (McGraw-Hill, 1973)

Office Planning System (Morganelli-Heumann Inc., 1976)

N. Okino et al., "TIPS-1: Technical Information Processing Systems for Computer Aided Design," Prolamat 73 Proceedings, Budapest (1973)

OXSYS-BOS: A Short Technical Description (Applied Research of Cambridge Ltd., May 1976)

PADL Primer, Production Automation Project (University of Rochester, July 1976)

C. Paul, LUMIS System: Final Report (Jet Propulsion Laboratory, 1976)

R. P. G. Pennington, "Computerized Product Selection by Performance," Architecture Canada (June 1967, pp 33-37)

G. J. Peters, "Interactive Computer Graphics Application of the Parametric Bi-Cubic Surface to Engineering Design Problems," in R. E. Barnhill and R. F. Riesenfeld (eds.), Computer Aided Geometric Design (Academic Press, 1974)

T. K. Peucker and N. Chrisman, "Cartographic Data Structures," The American Cartographer, Vol 2, No. 1 (April 1975)

P. A. Purcell, "Computer-Aided Architecture in the United Kingdom," in N. Negroponte (ed.), Computer Aids to Design and Architecture (Petrocelli-Charter, 1975)

P. Richens, "Geometry and Numbers in Building Systems," in D. Hawkes (ed.), Models and Systems in Architecture and Building (The Construction Press, 1975)

P. Richens, OXSYS-BDS User's Manual (Applied Research of Cambridge Ltd., February 1977)

T. R. Rhodes, "The Computer-Aided Design Environment Project (COMRADE)," Proceedings of the National Computer Conference (1973)

J. K. Robertson, "A Graphics System for Buidling Design in the Canadian Government," CAD 76 (IPC Science and Technology Press, 1976)

D. Roos et al., ICES System General Description, MIT Department of Civil Engineering Report R67-49 (September 1967)

D. T. Ross, The AED Approach to Generalized Computer Aided Design, MIT Report ESL-R-305 (MIT, 1967)

D. T. Ross, "The AED Free Storage Package," Communications of the ACM, Vol 10, No. 8 (August 1967)

D. T. Ross and J. W. Bracket, Automated Engineering Design (AED) Used for Graphics (Softech, undated)

M. A. Sabin (ed.), Programming Techniques in Computer Aided Design (NCC Publications, 1974)

J. Sammet, Programming Languages (Prentice-Hall) (This survey is periodically updated by articles in the Communications of the Association for Computing Machinery, most recently in 1977)

The Software Development Group, User Manual for the Surface Display Library (Dynamic Graphics Inc., April 1976)

G. Spur, J. Gausemeier, and G. Muller, COMPAC: The Use of Computer Internal Workpiece Models for Design and Manufacturing (Technical University of Berlin, 1976)

G. Stiny, Pictorial and Formal Aspects of Shape and Shape Grammars (Birkhaeuser-Verlag, Basel, 1975)

I. J. Sutherland, "Three Dimensional Data Input by Tablet," Proceedings of the IEEE, Vol 62, No. 64 (April 1976)

L. C. Teague, "Network Models of Configurations of Rectangular Parallelepipeds," in G. T. Moore (ed.) Emerging Methods in Environmental Design and Planning (MIT Press, 1970)

R. Thornton, MODEL: Interactive Modeling in Three Dimensions Through Two Dimensional Windows, Unpublished MS Thesis (Cornell University, 1976)

K. A. Vanlehn, "SAIL User Manual," Stanford Computer Sciences Reports STAN-CS-73-373 (July 1973)

A. van Wijngaarden et al., Revised Report on the Algorithmic Language ALGOL 68 (Springer-Verlag, Heidelberg, 1976)

View 3-D Package (Decision Graphics, undated)

H. B. Voelcker et al., An Introduction to PADL Production Automation Project (University of Rochester, 1974)

H. Voelcker and A. Requicha, "Geometric Modelling of Mechanical Parts and Processes," Computer Science and Computer Engineering Research Review (University of Rochester, 1976/77)

R. D. Warrender, "GENESYS: A Group of Papers," Computer Languages for Building, CIB Symposium by Correspondence CIB W52, Budapest (1975)

R. M. Wehrli et al., ARCAID: The Architect's Computer Graphics Aid (University of Utah, Department of Architecture, September 1969)

R. Weiss, "BE-VISION," Journal of the ACM, Vol 13, No. 2

G. Weinzapfel and S. Handel, "IMAGE: Computer Assistant for Architectural Design," in C. Eastman (ed.), Spatial Synthesis in Computer Aided Building Design (Wiley, 1975)

J. Weizenbaum, "Symmetric List Processor," Communications of the ACM, Vol 6 (1963)

N. Wirth and H. Weber, "EULER: A Generalization of ALGOL, and its Formal Definition," Journal of the ACM, Vol 9, nos. 1 and 2 (January and February 1966)

P. A. Woodsford et al., GINO, CAD Group Document (Cambridge University, June 1969)

J. M. Yang and S. J. Fenves, Representation of Information in the Design-Construction Process, Department of Civil Engineering Report No. R74-1 (Carnegie-Mellon University, undated)

C. Yessios, "Formal Languages for Site Planning," in C. Eastman, (ed.), Spatial Synthesis in Computer Aided Building Design (Wiley, 1975)

CERL DISTRIBUTION

Picatinny Arsenal
ATTN: SMUPA-VP3

US Army, Europe
ATTN: AEAEN (2)

Director of Facilities Engineering
APO New York 09827

DARCOM STIT-EUR
APO New York 09710

West Point, NY 10996
ATTN: Dept of Mechanics
ATTN: Library

Chief of Engineers
ATTN: Tech Monitor
ATTN: DAEN-ASI-L (2)
ATTN: DAEN-FEB
ATTN: DAEN-FEM
ATTN: DAEN-FEP
ATTN: DAEN-FESA
ATTN: DAEN-FEZ-A
ATTN: DAEN-MCZ-S
ATTN: DAEN-RDL
ATTN: DAEN-RMS (12)
for forwarding to
National Defense Headquarters
Director General of Construction
Ottawa, Ontario K1A0K2
Canada

Canadian Forces Liaison Officer (4)
U.S. Army Mobility Equipment
Research and Development Command
Ft Belvoir, VA 22060

Div of Bldg Research
National Research Council
Montreal Road
Ottawa, Ontario, K1A0R6

Airports and Const. Services Dir.
Technical Information Reference
Centre
KA0L, Transport Canada Building
Place de Ville, Ottawa, Ontario
Canada, K1A0N8

British Liaison Officer (5)
U.S. Army Mobility Equipment
Research and Development Center
Ft Belvoir, VA 22060

Ft Belvoir, VA 22060
ATTN: ATSE-TD-TL (2)
ATTN: Kingman Bldg, Library
ATTN: Learning Resources Center

Ft Monroe, VA 23651
ATTN: ATEN
ATTN: ATEN-C
ATTN: ATEN-FE-U

Ft Lee, VA 23801
ATTN: DRXMC-D (2)

Ft McPherson, GA 30330
ATTN: AFEN-FED

USA-WES
ATTN: Concrete Laboratory
ATTN: Library

6th, US Army
ATTN: AFKC-LG-E

US Army Engineer District
Saudi Arabia
ATTN: Library
New York
ATTN: Chief, Design Br.

US Army Engineer District
Pittsburgh
ATTN: Library
ATTN: Chief, Engr Div
Philadelphia
ATTN: Library
ATTN: Chief, NAPEN-D
Baltimore
ATTN: Library
ATTN: Chief, Engr Div
Norfolk
ATTN: Library
ATTN: Chief, NAOEN-D
Huntington
ATTN: Library
Wilmington
ATTN: Chief, SAWEN-PP
ATTN: Chief, SAWEN-PM
Charleston
ATTN: Chief, Engr Div
Savannah
ATTN: Library
ATTN: Chief, SASAS-L
Jacksonville
ATTN: Library
ATTN: Const. Div
ATTN: Env. Res. Br.
Mobile
ATTN: Library
ATTN: Chief, SAMEN-C
Nashville
ATTN: Library
Memphis
ATTN: Chief, LMMED-D
Louisville
ATTN: Chief, Engr Div
Detroit
ATTN: Library
ATTN: Chief, NCEED-T
St Paul
ATTN: Chief, ED-D
Chicago
ATTN: Chief, NCCVE
St Louis
ATTN: Library
ATTN: Chief, ED-D
Kansas City
ATTN: Library (2)
ATTN: Chief, Engr Div
Omaha
ATTN: Chief, Engr Div
New Orleans
ATTN: Chief, LMNED-DG
Little Rock
ATTN: Chief, Engr Div
Tulsa
ATTN: Library
Fort Worth
ATTN: Library
Albuquerque
ATTN: Library
ATTN: Chief, Engr Div
Los Angeles
ATTN: Library
ATTN: Chief, SPLED-D
San Francisco
ATTN: Chief, Engr Div
Sacramento
ATTN: Library, Room 8307
ATTN: Chief, SPKED-D
Far East
ATTN: Chief, Engr Div
Portland
ATTN: Library
ATTN: Chief, DB-6
Seattle
ATTN: Chief, EN-DB-ST
Walla Walla
ATTN: Library
ATTN: Chief, Engr Div
Alaska
ATTN: Library
ATTN: Chief, NPADE-R

US Army Engineer Division
Europe
ATTN: Technical Library
New England
ATTN: Library
ATTN: Chief, NEDED-T
North Atlantic
ATTN: Chief, NADEN-T
Middle East (Rear)
ATTN: MEDED-T
South Atlantic
ATTN: Chief, SADEN-TA
ATTN: Library
Huntsville
ATTN: Library (2)
ATTN: Chief, HNDED-CS
ATTN: Chief, HNDED-M
Lower Mississippi Valley
ATTN: Library
Ohio River
ATTN: Library
ATTN: Chief, Engr Div
North Central
ATTN: Library
ATTN: Chief, Engr Div
Missouri River
ATTN: Library (2)
Southwestern
ATTN: Library
ATTN: Chief, SWDED-TA
Pacific Ocean
ATTN: Chief, Engr Div
ATTN: FM&S Branch
ATTN: Chief, PODED-D
North Pacific
ATTN: Chief, Engr Div

Facilities Engineer
Ft Campbell, KY 42223
Ft Hood, TX 76544
FORSCOM
Ft Devens, MA 01433
Ft McPherson, GA 30330
Ft Lewis, WA 98433
TRADOC
Ft Dix, NJ 08640
Ft Gordon, GA 30905
Ft Knox, KY 40121
Ft Chaffee, AR 72905
Ft Sill, OK 73503
Ft Bliss, TX 79916
DSCPER
West Point, NY 10996
USATCFE
Ft Eustis, VA 23604
USAIC
Ft Benning, GA 31905
CAC&FL
Ft Leavenworth, KS 66027
AMC
Dugway, UT 84022
USACC
Ft Huachuca, AZ 85613
HQ, 1st Inf Div & Ft Riley, KS 66442
HQ, 5th Inf Div & Ft Polk, LA 71459
HQ, 7th Inf Div & Ft Ord, CA 93941

AF Civil Engr Center/XRL
Tyndall AFB, FL 32401

Naval Facilities Engr Command
Alexandria, VA 22332

Port Hueneme, CA 93043
ATTN: Library (Code L08A)
ATTN: Moreell Library

Washington, D.C.
ATTN: Building Research Advisory Board
ATTN: Library of Congress (2)

Defense Documentation Center (12)

Engineering Societies Library
New York, NY 10017

US Army Engineer Dist
Baltimore
ATTN: Enn Veskimets
Fort Worth
ATTN: Bobby Duvall
Mobile
ATTN: G. L. Phillips, Jr.
Omaha
ATTN: MRO/O. H. Asleson
Sacramento
ATTN: SPK/E. G. Jones
Kansas City
ATTN: MRK/P. Barber
New York
ATTN: NAN/W. Rodwick
Alaska
ATTN: NPA/R. Oenbrink
Far East
ATTN: POF/E. N. Moon
Japan
ATTN: POJ
Savannah
ATTN: SAS/L. Seals
Norfolk
ATTN: NAO/CPT B. Baccei

US Army Engineer Div
Pacific Ocean
ATTN: POD/C. Vandevelde
South Atlantic
ATTN: DAD/T. Nichols
Huntsville
ATTN: HND/J. Ammons
Europe
ATTN: EUD/R. N. Wheeler
Middle East
ATTN: Charles Thomas
North Atlantic
ATTN: NAD/T. C. Thompson
North Pacific
ATTN: NPD/U. Mettler
South Pacific
ATTN: SPD/J. O'Neill
Southwestern
ATTN: SWP/J. Miller

HQ TRADOC
ATTN: Ray Spunzo
Ft Monroe, VA 23651

HQ FORSCOM
ATTN: R. Collins
Ft McPherson, GA 30330

DARCOM, Installation & Services Activity
ATTN: DRCIS-RI/J. Parish
Rock Island Arsenal, IL 61201

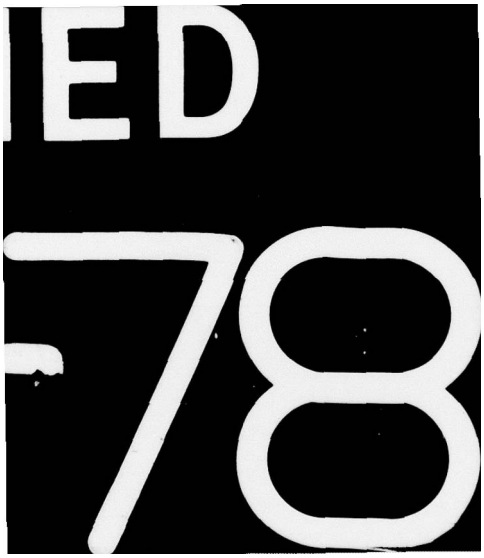
USA Materiel Dev & Readiness Command
ATTN: DRCIS-EF/C. A. Kuhn
5001 Eisenhower Ave
Alexandria, VA 22333

Commander
USA Air Defense Center
ATTN: ATZC-FE/J. Kemp
Ft Bliss, TX 79906

HQDA
DAEN-MCE/L. Garrett
DAEN-DFE/L. Blakey
DAEN-MCE/D. Butler
DAEN-MCE-D/V. Gottschalk
DAEN-MCE-P/C. Dodge
DAEN-MCE-S/R. Hatwell
DAEN-MCE-A/R. Shibley
DAEN-MCE-S/W. Darnell
DAEN-MCE-D/L. Schindler
DAEN-MCE-P/F. Beck
DAEN-MCE-D/E. Dudka
DAEN-DSE-R/McMurrer
DAEN-FEM-S/P. Sabo
DAEN-FEM/W. Ahloo
DAEN-RDM/J. Healy
DAEN-DSP/L. Wells
DAEN-MCZ-S/R. Wells
DAEN-FEB-S/R. Roberts
DAEN-RDM/LTC Newsom

Mitchell, William J
Computer representation of three-dimensional structures for CAEADS / by William J. Mitchell, Mary Oliverson. -- Champaign, Ill : Construction Engineering Research Laboratory ; Springfield, Va : available from National Technical Information Service , 1978.
195 p. : ill ; 27 cm. (Technical report - Construction Engineering Research Laboratory ; P-86)

1. Architectural design -- data processing. 2. CAEADS 3. Computer graphics. I. Oliverson, Mary. II. Title. III. Series: U. S. Construction Engineering Research Laboratory. Technical report ; P-86.


ED
78